VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA

FAKULTA STROJNÍ

Katedra automatizační techniky a řízení

Studijní opory základů programování na platformě .NET Micro Framework

Educational Support of Basics of Development on the .NET Micro Framework Platform

Vedoucí práce: Ing. David Fojtík, Ph.D.

Student: David Bielesz

Ostrava 2016

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání bakalářské práce

Student: **David Bielesz**

Studijní program: B2341 Strojírenství

Studijní obor: 3902R001 Aplikovaná informatika a řízení

Téma: Studijní opory základů programování na platformě .NET Micro Framework
Educational Support of Basics of Development on the .NET Micro Framework Platform

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Seznamte se s platformou .NET Micro Framework a aktuálním hardwarem podporující tuto technologii.
2. Navrhněte a vytvořte sadu demonstrujících úloh, které v několika krocích seznámí studenta s principy programování na platformě .NET Micro Framework a obsluhou vybraných standardních rozhraních.
3. Navržené příklady a použitá hardwarová rozhraní vhodně popište tak, aby vzniklé materiály mohly sloužit k samostudiu začátečníkům.
4. Zpracujte materiály do podoby studijních opor a zhodnoťte dosažené výsledky.

Seznam doporučené odborné literatury:

GHI ELECTRONICS *Beginners Guide to C# and the .NET Micro Framework.* GHI Electronics 2012, 58 p., PDF Avaliable from: <URL: http://www.ghielectronics.com/downloads/FEZ/Beginners%20guide%20to%20NETMF.pdf>.
GHI ELECTRONICS *.NET Micro Framework for Beginners.* GHI Electronics, LLC, November 12, 2015, 68 p., PDF, Avaliable from: <URL: https://www.ghielectronics.com/downloads/NETMF/NETMF_for_Beginners.pdf>.
KUČERA, J. *Microsoft .NET Micro Framework Tools & Resources.* dostupné z: <URL: http://informatix.miloush.net/microframework/Home.aspx>.
MALIN, J. R., LIMING, S. D. *Professional's Guide to .NET Micro Framework Application Development.* January 7, 2012, Annabooks;1 edition, ISBN: 978-0-9842801-9-3
MICROSOFT .NET Micro Framework Platform SDK dostupné z WWW: <URL: http://msdn.microsoft.com/en-us/library/ee436350.aspx>.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Fojtík, Ph.D.**

Datum zadání: 11.12.2015

Datum odevzdání: 16.05.2016

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta strojní 3 OSTRAVA

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

doc. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

Místopřísežné prohlášení studenta

Prohlašuji, že jsem celou bakalářskou práci včetně příloh vypracoval samostatně pod vedením vedoucího bakalářské práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě 13. 5. 2016

.....

podpis studenta

Prohlašuji, že

• jsem byl seznámen s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo.

• beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo nevýdělečně ke své vnitřní potřebě bakalářskou práci užít (§ 35 odst. 3).

• souhlasím s tím, že bakalářská práce bude v elektronické podobě uložena v Ústřední knihovně VŠB-TUO k nahlédnutí a jeden výtisk bude uložen u vedoucího bakalářské práce. Souhlasím s tím, že údaje o kvalifikační práci budou zveřejněny v informačním systému VŠB-TUO.

• bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona.

• bylo sjednáno, že užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

• beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ostravě 13. 5. 2016

.....Bielesz.....

podpis

Jméno a příjmení autora práce: David Bielesz

Adresa trvalého pobytu autora práce: Ostrava, Poruba, Bedřicha Nikodéma 4482/7, 708 00

BIELESZ D. *Studijní opory základů programování na platformě .NET Micro Framework*. Ostrava: kat. ATŘ – 352 VŠB-TU, 2016. 26 str. Bakalářská práce, vedoucí: Fojtík, D.

V bakalářské práci je popsána funkce a možnosti platformy .NET Micro Framework, přehled hardwaru pro její využití a demonstrační úlohy.

Hlavní náplní bylo detailní seznámení se s platformou .NET Micro Framework, s dostupným hardwarem a se základním programováním.

Výsledkem práce jsou materiály, s jejichž pomocí se lze seznámit s platformou .NET Micro Framework a s aktuálně dostupným hardwarem pro její využití. Rovněž jsou k dispozici vypracované úlohy pro vybraný hardware, jež byl dostupný.

Klíčová slova: .NET Micro Framework, .NET MF, Studijní opory, FEZ Cerbuino Net, STM32 F429 DISCOVERY

ANOTATION OF THESIS

BIELESZ D. *Educational Support of Basics of Development on the .NET Micro Framework Platform*. Ostrava: kat. ATŘ – 352 VŠB-TU, 2016. 26 pages. Bachelor Thesis, Supervisor of project: Fojtík, D.

Bachelor Thesis decribes main functions and features of .NET Micro Framework platform, hardware overview for platform and demonstration tasks.

The main focus of the work was on detailed familiarization with .NET Micro Framework platform, with available hardware and with basic programming.

Results of the work are materials which help getting to know basics of .NET Micro Frameework platform and currently available hardware for its usage. There are also available examplary tasks for selected harware.

Keywords: .NET Micro Framework, .NET MF, Study materials, FEZ Cerbuino Net, STM32 F429 DISCOVERY

Obsah

Seznam použitých zkratek

ARM je v informatice označení architektury procesorů používaných díky své nízké spotřebě elektrické energie zejména v mobilních zařízeních (mobilní telefony, tablety).

CLR (Common Language Runtime) – součást virtuálního počítače .NET Frameworku Microsoftu, která řídí vykonání .NET programů.

EEPROM (Electrically Erasable Programmable Read-Only Memory) – jedná se o elektricky mazatelnou semipermanentní (nevolatilní) paměť typu ROM-RAM.

GPIO (General-purpose input/output) – je obecný pin na integrovaném obvodu, jehož chování, včetně toho, zda se jedná o vstupní nebo výstupní pin, je ovladatelné uživatelem za běhu.

HAL (Hardware abstraction layer) – je v informatice hardwarová abstrakční vrstva, která v počítači vytváří jednotné rozhraní (API) ovládající různě fungující hardware.

I²C (Inter-Integrated Circuit) – je multi-masterová počítačová sériová sběrnice vyvinutá firmou Philips, která je používána k připojování nízko rychlostních periferií k základní desce, vestavěnému systému nebo mobilnímu telefonu.

PAL (Platform Abstraction Layer) – umožňuje vytvořit referenční design z jednoduchého souboru popisujícího desku.

PWM (Pulse Width Modulation) – je diskrétní modulace pro přenos analogového signálu pomocí dvouhodnotového signálu.

RAM (Random-Access Memory) – je v informatice typ paměti, u níž je libovolné paměťové místo přístupné za stejnou vybavovací dobu.

SBC (session border controller) – je zařízení pravidelně využívané ve Voice over Internet Protocol sítích (VoIP), zde provádí kontrolu nad signalizací a obvykle také toky médií podílejících se na zakládání, vedení, a ukončení telefonních hovorů nebo jiné interaktivních mediální komunikace.

SDK (Software development kit) – je typická sada vývojových nástrojů umožňující vytváření aplikací frameworky.

SOAP (Simple Object Access Protocol) – je protokolem pro výměnu zpráv založených na XML přes síť, hlavně pomocí HTTP.

SPI (Serial Peripheral Interface) – je sériové periferní rozhraní.

TCP/IP (Transmission Control Protocol/Internet Protocol) – obsahuje sadu protokolů pro komunikaci v počítačové síti a je hlavním protokolem celosvětové sítě Internet.

WCF (Windows Communication Foundation) –) je sada knihoven, API a běhové prostředí, dohromady tvořící framework v rámci frameworku .NET, zajišťující komunikaci mezi aplikacemi a umožňující vytvářet servisně orientované aplikace.

WPF (Windows Presentation Foundation) – je podmnožinou .NET Frameworku od verze 3.0, který používá značkovací jazyk XAML pro vytvoření "uživatelsky bohatého rozhraní" (RUI).

WSDL (Web Services Description Language) – popisuje, co nabízí webová služba za funkce a způsob, jak se jí na to zeptat.

1 Úvod

Následující práce se bude zabývat zejména úvodem do .NET Micro Framework, jeho využití a vývojem. .NET Micro Framework se potýká zejména s problémem, že není oficiálně příliš podporován a je tlačen kupředu především komunitou, jež objevila jeho potenciál, než přímo samotnými vývojáři .NET Micro Framework.

Tímto tématem jsem se rozhodl ve své práci zabývat, jelikož jsem měl zájem o rozšíření svých znalostí v oblasti programování a zároveň se i seznámit s touto platformou, která má potenciál bohatého využití v oblasti výuky i praktických aplikací.

Pro zahájení práce s platformou .NET Micro Framework je zapotřebí projít řadu materiálů, protože kromě samotného seznámení se s platformou je rovněž zapotřebí prostudovat hardwarová specifika daných procesorových desek a zároveň se seznámit s programováním v jazyce C# nebo Visual Basic, v případě, že není k dispozici předchozí zkušenost.

Zpracování do podoby studijních materiálů představuje rozšíření zatím nevelkého množství podkladů pro práci s touto platformou, což komplikuje situaci pro ty, kteří se rozhodnou sami vzdělávat v této oblasti bez předchozích základů. Materiály jsou zaměřeny na to, aby byly stručné a srozumitelné.

2 Seznámení s .NET Micro Framework

.NET Micro Framework je vývojové a realizační prostředí pro zařízení, jež operuje s omezenými zdroji, původně vyvinuté v Microsoft Startup Business Accelerator, ale nedávno bylo přesunuto do vývojářské divize, aby se přiblížilo celkovému směru, jímž se ubírají vývojářské snahy Microsoft [Galli 2009].

Výsledkem tohoto je, že .NET Micro Framework se stal bezproblémovou vývojářskou zkušeností přinášející jediný programovací model a nástrojový řetězec pro široký rámec vývojářských řešení a to v rozsahu od malých inteligentních zařízení po servery a cloud. Rovněž už nadále nejsou časově omezené verze [Galli 2009].

Zahrnutí zdrojové kódy pro téměř všechny produkty rovněž zajišťuje, že vývojáři mají nyní přístup ke knihovnám základních tříd, jež byly implementovány pro .NET Micro Framework a samotný CLR kód [Galli 2009].

Avšak oba TCP/IP stack a kryptografické knihovny nejsou zahrnuty ve zdrojovém kódu. Je tomu tak proto, že TCP/IP stack je software patřící třetí straně a Microsoft má k němu licenci od EBSNet, takže nejsou k dispozici práva pro distribuci tohoto zdrojového kódu. Pokud potřebuje přístup ke zdrojovému kódu pro TCP/IP stack, je nutné kontaktovat přímo EBSNet [Galli 2009].

Co se týče kryptografických knihoven, ty nejsou součástí zdrojového kódu, jelikož mají využití i mimo rámec .NET Micro Framework. Zákazníci, kteří potřebují získat přístup ke kódu v kryptografických funkcích, zjistí, že tyto knihovny lze nahradit [Galli 2009].

Microsoft má v úmyslu nadále se aktivně věnovat probíhajícímu vývoji .NET Micro Framework a pracovat společně s komunitou. Ačkoli licence umožní zákazníkům vzít kód a vytvořit specializované verze, jež naplňují jejich představy a potřeby, bylo z jejich strany dáno jasně najevo, že si přejí, aby Microsoft zůstal zapojen ve vývoji, aby se předešlo potenciálnímu fragmentování platforem [Galli 2009].

Vzhledem k těmto okolnostem je v plánu vytvoření hlavního technologického týmu, jež se bude skládat jak ze zaměstnanců Microsoft, tak z lidí, jež nejsou jeho součástí. Hlavním úkolem týmu bude pokračovat ve vývoji vysoce kvalitních produktů pro velice malá zařízení. Tato skupina bude zastávat úkol prostředníka s komunitními vývojáři a zároveň umožní vývojářům Microsoft, aby pokračovali ve zvyšování funkcionality a koordinaci

s celým .NET týmem. Stránka bude rovněž podporovat lidi, vytvářející rozšíření, které budou existovat souběžně s platformou namísto přímé integraci do ní.

Microsoft .NET Micro Framework kombinuje spolehlivost a efektivitu spravovaného kódu s vývojovými nástroji Microsoft Visual Studio®, aby bylo možno zaručit výjimečnou efektivitu nezbytnou k vývoji zabudovaných aplikací pro malá zařízení. Microsoft .NET Micro Framework SDK podporuje vývoj kódu, a to včetně I/O zařízení, v.NET Micro Framework (NETMF) je open-source .NET platforma pro zařízení se značně omezenými zdrojovými možnostmi, mají-li alespoň 256 KBytes flash a 64 KBytes RAM. To zahrnuje malé verze .NET CLR a podporuje vývoj v C#, Visual Basic .NET a debugování (při emulaci nebo přímo na hardware) a je plně integrován ve vývojovém prostředí Microsoft Visual Studio®. NETMF obsahuje sestavu knihoven základních tříd .NET (kolme 70 tříd s přibližně 420 metodami), implementace Windows Communication Foundation (WCF), a GUI frameworku, volně vycházejícím z Windows Presentation Foundation (WPF) a Web Services stack založeném na SOAP a WSDL. NETMF rovněž v sobě zahrnuje řadu dalších knihoven využitých u vestavěných aplikací [.NET Micro Framework].

.NET Micro Framework Interpreter

.NET Micro Framework interpreter je projekt obsahující zdrojový kód pro .NET Micro Framework interpreter, knihovny základních tříd a vzorky pro portování Hardware Abstraction a Platform Abstraction layers (HAL/PAL) [.NET Micro Framework].

2.2 Vydání

Významné projekty usnadňující práci s .NET Micro Framework:

- SDK v4.3 (QFE2-RTM)

  Pro vývoj aplikací v .NET Micro Framework

- Porting Kit 4.3 (RTM QFE1)

  Pro vytváření zařízení, na kterých poběží .NET Micro Framework

.NET Micro Framework se zaměřuje na vestavěný vývoj, jež je snazší, rychlejší a rovněž také levnější, tím, že vývojářům v této oblasti nabízí přístup k moderním technologiím a

nástrojům, které jsou obvykle využívány desktop vývojáři aplikací. Navíc dává možnost desktop .NET vývojářům využít jejich dovednosti ve světě vestavěných aplikací, čímž se také zvyšuje množství vývojářů kvalifikovaných pro práci v této oblasti [.NET Micro Framework].

.NET Micro Framework je součást .NET Foundation. Na konferenci Build 2014 bylo oznámeno, že .NET Foundation byl vytvořen jako nezávislé fórum, které má za cíl podpořit otevřený vývoj a spolupráci v rozrůstajícím se kolektivu open-source technologií pro .NET [.NET Micro Framework].

2.3 Funkce

Mezi unikátní funkce .NET Micro Framework (srovnáme-li jej s ostatním .NET platformami) patří:

- Paměťová stopa kolem 300 KB; pro srovnání, další nejmenší .NET implementace,.NET Compact Framework fungující na Windows CE, vyžaduje přibližně 12 MB.
- Možnost běžet přímo bez jakéhokoli operačního systému, zatímco simultánně je tato možnost zachována.
- Podporuje běžná vestavěné periferie a propojení, včetně flash paměti, EEPROM, GPIO, I²C, SPI, sériového portu, USB.
- Optimalizace pro efektivní nakládání s energií v baterii napájených zařízeních.
- Nevyžaduje žádnou jednotku pro správu paměti.
- Poskytuje podporu pro multithreading i v případě provozu na zařízeních se single-thread operačním systémem.
- Hardware abstraction layer umožňuje portování k dalším architekturám.
- Spravovaný model ovladače zařízení poskytuje možnost psaní ovladačů pro mnoho různých zařízení v C#.
- Omezení provedení, aby bylo zabráněno zablokování a pádům zařízení.
- Transparentní podpora pro ukládání objektů v energeticky nezávislé paměti.

Kvůli omezením, ve kterých pracuje, je .NET Micro Framework omezen více, než jen svými zjednodušenými knihovnami. Na příklad tato platforma nepodporuje symertický multi-procesing, více rozměrová pole, strojově závislé typy nebo nebezpečné instrukce. CLR je tlumočníkem spíše než just-in-time kompilátorem a využívá jednodušší mark-and-sweep

garbage collector namísto generačního přístupu. Součinnost mezi spravovaným a přirozeným kódem má v současnosti celou řadu omezení. V současnosti .NET Micro Framework nepodporuje žádné jiné .NET jazyky než C# a Visual Basic [.NET Micro Framework].

2.4 Podpora

.NET Micro Framework je současně podporován na ARM architektuře procesorů (včetně ARM7, ARM9 a Cortex-M architektur) a v minulosti byl rovněž podporován na analogových zařízeních Blackfin. Porting Kit je nyní dostupný zdarma ke stažení společně se zdrojovým kódem pod licencí Apache 2.0 v Microsoft Download Center [.NET Micro Framework].

.NET Micro Framework má kořeny v SPOT iniciativě Microsoftu a byl použit v MSN Direct produktech, jako jsou třeba chytré hodinky před tím, než byly zpřístupněny vývojářům z třetích stran začátkem roku 2007. Je rozšířenou platformou pro Windows SideShow zařízení a byl adoptován dalšími trhy a sice hospodaření s energií, zdravotnictví, průmyslová automatizace a senzorové sítě [.NET Micro Framework].

Microsoft umožňuje vývojářům vytvářet aplikace s využitím .NET Micro Framework bez poplatků a zpřístupňuje zdarma ke stažení SDK, který lze používat se všemi verzemi Visual Studio a to včetně zdarma dostupných expresních edicí.

2.5 Historie

- V listopadu 2009 Microsoft vydal zdrojový kód framework pro vývojářskou komunitu jako součást Apache 2.0 licence.
- V lednu 2010 Microsoft odstartoval netmf.com, stránku pro vývojářskou komunitu, aby byla umožněna koordinace probíhajícího vývoje základních implementací s open-source komunitou.
- 9. ledna 2010 GHI Electronics oznámili FEZ Domino, což je první člen řady produktů nazvané FEZ (Freakin' Easy!). Jedná se o kombinaci open-source hardware s patentovanou closed-source verzí .NET Micro Framework.
- 3. Srpna 2010 Secret Labs oznámil Netduino představující 100% open-source elektronickou platformu využívající .NET Micro Framework.

- V únoru 2011 Novell poskytl první náhled na Mono 2.12 C# kompilátor, ten je první open-source kompilátor pro .NET Micro Framework.

2.6 Hardware

Celá řada prodejců vyrábí čipy, vývojářské soupravy a další platformy, na kterých funguje .NET Micro Framework:

- Netduino od Secret Labs
  Netduino je open-source elektronická platforma využívající .NET Micro Framework.
- GHI Electronics
  GHI Electronics vyrábí několik modulů .NET Micro Framework.
- EMX Module.
- ChipworkX Module
- USBizi144 Chipset a USBizi100, které se od sebe liší jedině tím, že USBizi100 postrádá podporu USB host.

GHI Electronics rovněž vytváří ".NET FEZ" řadu velice malých open-source hardware desek s patentovaným firmware, které jsou zaměřeny pro začátečníky. Zakládají se na USBizi chipset a jejich funkcích. FEZ Domino deska nabízí USB host. Přestože FEZ je primárně zaměřen pro začátečníky, představuje rovněž velice levný startovní bod pro profesionály, jež mají v úmyslu prozkumat možnosti NETMF (.NET Micro Framework). Některé z těchto desek jsou fyzicky kompatibilní s Arduino [.NET Micro Framework].

Mountaineer Boards

Mountaineer Boards, součást Mountaineer Group, vytváří malý rozsah open-source open-hardware desek, které využívají .NET Micro Framework. Mountaineer portoval .NET Micro Framework pro využití v STM32 rodině mirkrokontrolerů fungujících na jejich Mountaineer deskách a i jinde [.NET Micro Framework].

STMicroelectronics

STMicroelectronics, tvůrce STM32 rodiny mirkrokontrolerů, vytvářejí levné discovery desky pro prezentaci kontrolerů. STMicroelectronics poskytují porty pro fungování .NET Micro Framework [.NET Micro Framework].

Netmfdevices

Netmfdevices je open-source elektronická platforma využívající FEZHacker a .NET Micro Framework [.NET Micro Framework].

Micromint

Micromint Bambino 200 je první multi-jádrový SBC kompatibilní s .NET Gadgeteer framework. Micromint Bambino 200 je poháněn NXP LPC4330, první dual-core ARM Cortex-M mikro kontrolérem. Jeho Cortex-M4 a Cortex-M0 jádra jsou obě schopna dosáhnout 204 MHz. S 264 KB SRAM desky a 4 MB flash, mohou vývojáři dosáhnout vysokých požadavků v monitorování, instrumentace, získávání dat, kontrola procesů a mnoha dalších aplikacích. Bambino 200E má všechny funkce totožné s Bambino 200, ale navíc přináší zvýšení flash paměti to 8 MB, 10 Gadgeteer patic, Ethernet port, microSD patici, a další funkce [.NET Micro Framework].

2.7 Možnosti NETMF

Z vývojářského hlediska má práce s NETMF dvě části. Za prvé portování do hardware a využití k ovládání právě tohoto hardware. Portování má značnou cenu, vyžaduje množství času a rovněž mnoho zkušeností. Poté ovšem využití NETMF je velice nenáročné.

S cílem dosáhnout co největší kvality a spolehlivosti byl NETMF obohacen o další příslušenství jako je podpora WiFi, USB Host a databází.

Dříve platforma podporovala pouze C# ale s vydáním .NET Micro framework v4.2 Beta přišla podpora Visual Basic pro .NET Micro-framework. Tato podpora vyžaduje instalaci Visual Studio 2010 SP. Podpora VB využívá práci vykonávanou 'Compilers' týmem pro VBCore funkci a oživuje další platformu [.NET Micro Framework].

Dodavatelé dalších SDK budou moci zdokonalit své SDK a poskytovat projektové šablony specifické pro VB zaměřené na verzi 4.2, kdy Secret Labs už vydávají 4.2 beta verzi svého SDK pro zařízení Netduino s podporou VB.

Toto komukoli umožňuje využití schopností nabytých ve VB v oblasti programování jejich vlastních zabudovaných zařízení. S tímto se vývojáři ve VB mohou stát u vývojáři mikro kontrolérů [.NET Micro Framework].

2.8 Gadgeteer

Microsoft .NET Gadgeteer je rapidní prototypující platforma pro malá elektronická zařízení a zabudovaná hardwarová zařízení. Kombinuje výhody objektově orientovaného programování, nenapájené soustavy elektroniky a soupravy hardwarových modulů, společně s rychlým zhotovením fyzického zastřešení s využitím počítačových návrhů [Microsoft 2015].

O .NET Gadgeteer

.NET Gadgeteer byl vytvořen výzkumníky z Microsoft, jako interní prototypující nástroj, ale z důvodu vnějšího zájmu zejména od lidí, kteří se tomuto věnovali ve volném čase nebo jako součást výuky, došlo ke změně na open source software, který má v současnosti k dispozici živý ekosystém harwaru od různých výrobců.

Platforma je postavena na .NET Micro Framework, což umožňuje malým zařízením aby byly programovány v jazyku c# a využili tak debugovacích nástrojů Visual Studia.

Jednotlivé .NET Gadgeteer moduly mohou být snadno propojeny navzájem, aby daly vzniknout zařízením jež jsou zároveň jednoduchá a sofistikovaná. Každý modul přidává nějaké nové možnosti, jako je třeba schopnost zobrazovat obrázky, přehrávat zvuky nebo umožnit uživatelskou interakci. Tato silná kombinace umožňuje, aby plně funkční zařízení byly prototypovány v rámci hodin a ne dnů, nebo dokonce týdnů [Microsoft 2015].

Nový způsob tvorby sofistikovaných zařízení

I lidé s minimem nebo dokonce žádnými základy v elektronice mohou vytvořit zařízení sestávající s komponentů, jako jsou senzory, světla, spínače, displeje, komunikační moduly, ovladače motoru a mnoho dalších. Stačí si vybrat komponenty, zapojit je do základní desky a naprogramovat je, aby dokázaly spolupracovat. .NET Gadgeteer využívá .NET Micro Framework, aby psaní kódu pro zařízení bylo co nejsnazší [Microsoft 2015].

Vzdělávání

.NET Gadgeteer je vhodný způsob pro seznámení studentů s programováním, elekronikou a rovněž i designem. .NET Gadgeteer lze využít, aby u lidí, kteří o to mají zájem, byly vyvinuty základní programovací schopnosti. Pro toto jsou k dispozici řady materiálů online jak pro samouky, tak i instruktory [Microsoft 2015].

.NET Gadgeteer Hardware

Systém .NET Gadgeteer je složen ze základní desky obsahující zabudovaný procesor a celou řadu modulů, které jsou napojeny k této základní desce jednoduchým plug-and-play rozhraním. K dispozici dnes existuje nepřeberné množství dostupných .NET Gadgeteer modulů zahrnující: displej, fotoaparát, síťování, ukládání a rozličné senzory a vstupní ovládací prvky. Současně nepřetržitě probíhají návrhy nových modulů, aby bylo pokryto co největší množství požadavků. Objímky základních desek .NET Gadgeteer jsou početné a každá z nich je poznačena jedním nebo více písmeny, které indikují, jaký typ modulu lze do ní zapojit [Microsoft 2015].

.NET Gadgeteer Software

.NET Gadgeteer zařízení jsou programovány v jazyku C# za využití .NET Micro Framework. Není problém aplikovat znalosti programování .NET na desktopu, webu nebo zabudovaných zařízeních. Intuitivní vizuální designer umožňuje rychlé a zároveň snadné odstartování projektů. Rovněž je k dispozici přístup k velice užitečnému příslušenství Visual Studia, jako je třeba Intellisense. Lze rovněž debugovta .NET Gadgeteer programy, zatímco jsou spuštěny na zařízení, cože je značně unikátní možnost, která dělá řešení problémů se zařízením .NET Gadgeteer do značné míry snazší [Microsoft 2015].

Fyzický návrh .NET Gadgeteer

Kombinace možností rapidního softwarového a hardwarového prototypování s inovativními nástroji fyzické konstrukce zařízení, jako představují na příklad 3D tiskárny nebo laserové řezačky. Celá řada vyvíjených nástrojů usnadní návrhy krabiček pro vytvářený .NET Gadgeteer projekty [Microsoft 2015].

3 Na trhu dostupné procesorové desky

Pro práci využiji data ze stránky www.shopmicroframework.eu, tato stránka pokrývá všechny u nás dostupné procesorové desky a rovněž poskytuje základní informace pro jejich popis. Podrobněji se budu věnovat popisu mnou využitých procesorových desek v následující části práce. Část desek je v době psaní této práce stále ještě dostupná, ale v roce 2016 již mají být nahrazeny novějšími verzemi, proto zde zařazeny nejsou.

Argon R1 - Gadgeteer Mainboard

USB zařízení, USB host, 3 sériové porty, 3 PWM výstupy, 4 analogové vstupy, procesor: 120Mhz LPC1788 Cortex-M3, RAM: 32MB, FLASH: 128MB, objímky: 14

FEZ Cerberus Mainboard

Procesor: 168Mhz 32bit Cortex M4, FLASH: 1MB, RAM: 192KB, USB host (bez podpory), USB zařízení, 8 .NET Gadgeteer kompatibilních objímek, 9 analogových vstupů, 2 analogové výstupy

FEZ Cerbuino Bee

Procesor: 168Mhz 32bit Cortex M4, FLASH: 1MB, RAM: 192KB, USB host, USB zařízení, 3 .NET Gadgeteer kompatibilní objímky, 9 analogových vstupů, 2 analogové výstupy

FEZ Cobra II (WiFi w/ ANT)

Procesor: 120Mhz 32bit ARM Cortex M3, FLASH: 2,87 MB, RAM: 13,67 MB, USB host, USB klient, 6 .NET Gadgeteer kompatibilních objímek, 8 analogových vstupů, analogový výstup

FEZ Hydra+ Mainboard

Procesor: 240Mhz ARM9, FLASH: 1,28 MB, RAM: 10 MB, USB klient, 14 .NET Gadgeteer kompatibilních objímek, 6 analogových vstupů

FEZ Lemur

Procesor: 84 MHz 32-bit ARM Cortex-M4, FLASH:128 KB, RAM: 64 KB, USB klient, 16 analogových vstupů

FEZ Panda III

Procesor: 180 MHz 32-bit ARM Cortex-M4, FLASH: 256 KB, RAM: 156 KB, USB klient, USB host, 16 analogových vstupů, 2 analogové výstupy

FEZ Raptor Mainboard

Procesor: 400 MHz 32-bit ARM 9, FLASH: 1.4 MB, RAM 92 MB, USB klient, USB host, 18 .NET Gadgeteer kompatibilních objímek, 12 analogových vstupů

FEZ Reaper Mainboard

Procesor: 180 MHz 32-bit ARM Cortex-M4, FLASH: 256 KB, RAM 156 KB, USB klient, USB host, 14 .NET Gadgeteer kompatibilních objímek, 12 analogových vstupů, 2 analogové výstupy

FEZ Spider II Mainboard

Procesor: 120 MHz 32-bit ARM Cortex-M3, FLASH: 2.87 MB, RAM 13.67 MB, USB klient, USB host, 14 .NET Gadgeteer kompatibilních objímek, 6 analogových vstupů, 1 analogový výstup

Mountaineer Ethernet Mainboard

Procesor: STM32F407, operační frekvence: 168 MHz/210 DMIPS, FLASH: 1 MB (čip) + 8 MB (deska), RAM: 192 KB (čip), 8 .NET Gadgeteer kompatibilních objímek

Mountaineer USB Mainboard (nejmenší na trhu)

Procesor: STM32F407, operační frekvence: 168 MHz/210 DMIPS, FLASH: 1 MB (čip) + 8 MB (deska), RAM: 192 KB (čip), 9 .NET Gadgeteer kompatibilních objímek

NANO Mainboard

Procesor: 200 Mhz ARM 9, FLASH: 8 MB, DRAM: 8 MB, USB zařízení, 10 .NET Gadgeteer kompatibilních objímek

4 Popis STM32 F429 DISCOVERY a FEZ Cerbuino Net

Následně se budu věnovat bližšímu popisu STM32 F429 DISCOVERY a FEZ Cerbuino Net, které byly poskytnuty pro zpracování této práce, na ně rovněž budou vypracovány jednoduché demonstrační úlohy v rámci tohoto projektu. V popisu budou parametry a rovněž i schémata zařízení.

4.1 STM32 F429 DISCOVERY

STM32 F4 DISCOVERY pomáhá objevit STM32F407/417 řadu příslušenství a snadno vyvinout vlastní aplikace. Zahrnuje vše potřebné pro začátečníky i zkušené uživatele, aby mohli rychle začít. Založeno na STM32F407VGT6 zahrnuje zabudovaný debugovací nástroj ST-LINK/V2, dva ST MEMS digitální akcelerometry a digitální mikrofon, jeden DAC s integrovaným ovladačem reproduktorů třídy D, LED a tlačítky a USB OTG mikro-AB konektor.

Příslušenství

• STM32F407VGT6 mikro kontrolér zahrnující 32-bitový ARM Cortex®-M4 s FPU jádrem, 1 MB Flash paměti, 192 KB RAM v LQFP100 balíku

• palubní ST-LINK/V2 s přepínačem režimů výběru pro využití soupravy jako samostatného STLINK/V2 (s SWD konektorem pro programování a debugování)

• Palubní zdroj energie: skrz USB bus nebo externího V napájení

• Externí aplikace napájení: 3 V a 5 V

• LIS302DL nebo LIS3DSH ST MEMS 3-osý akcelerometr

• MP45DT02 ST MEMS audio senzor všesměrový digitální mikrofon

• CS43L22 audio DAC s integrovaným ovladačem reproduktorů třídy D

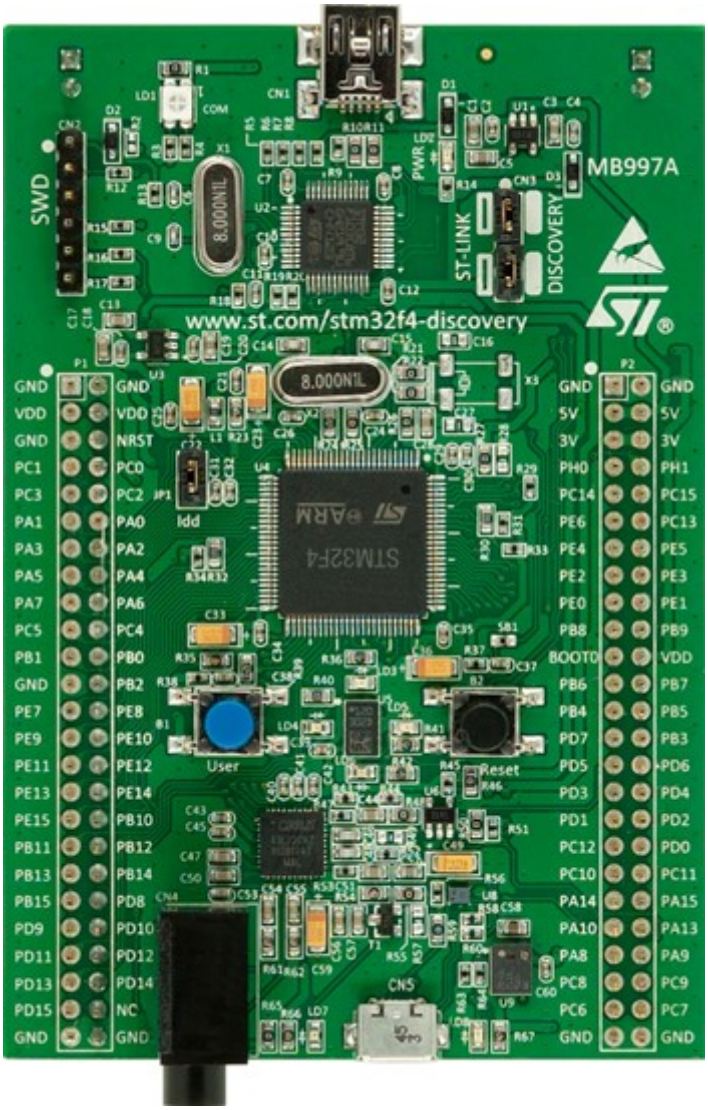
• Osm LED: – LD1 (červená/zelená) pro komunikaci USB – LD2 (červená) pro 3.3 V napájení – čtyři uživatelské LED, LD3 (oranžová), LD4 (zelená), LD5 (červená) and LD6 (modrá) – 2 USB OTG LEDs LD7 (zelená) VBus a LD8 (červená) nadměrný proud

• Dvě tlačítka (uživatel a reset)

• USB OTG FS s mikro-AB konektorem

• Rozšiřující hlavička pro všechny LQFP100 I/Os pro rychlé připojení k prototypující desce a snadný průzkum

• Obsáhlý bezplatný software zahrnující rozličné příklady, součást STM32CubeF4 balíčku nebo STSW-STM32068 pro využití dědičných standardních knihoven

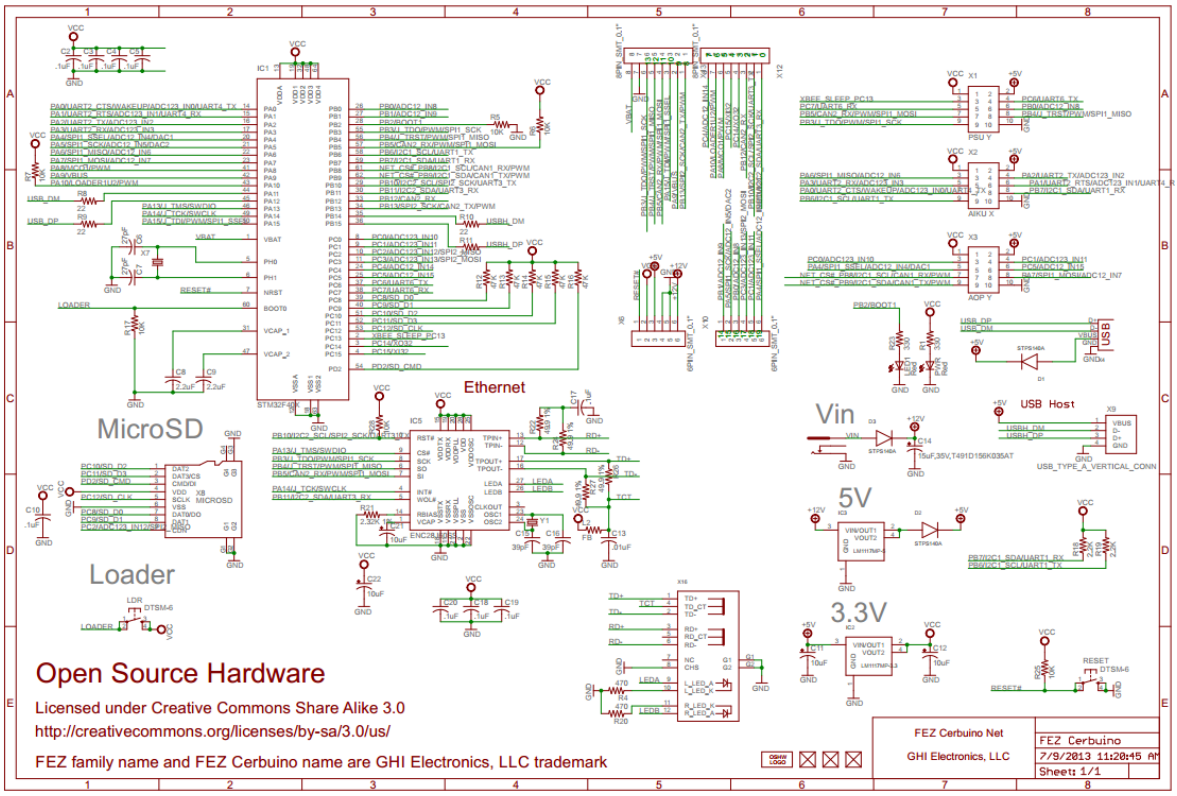


Obrázek 1 – STM32 F429 DISCOVERY [STMicroelectronics 2014]

4.2 FEZ Cerbuino Net

FEZ Cerbuino Net je Arduino-kompatibilní .NET Gadgeteer základní deska, na které běží .NET Micro Framework, což umožňuje uživatelům programovat desky z Visual Studio Microsoftu za využití jazyků C# nebo Visual Basic, což umožňuje vývojářům využít výhod rozsáhlých zabudovaných knihoven pro sítě, souborové systémy, grafické rozhraní a periferní zařízení.

Schéma:



Obrázek 2 – Schéma FEZ Cerbuino Net [GHI Electronics]

Tabulka 1 – Parametry FEZ Cerbuino Net [GHI Electronics]	
Processor	168 Mhz 32-bit Cortex-M4
Core Systém Hardware	Cerberus Chipset
Systém Platform	.NET Micro Framework
User Available Flash	384 KB
User Available RAM	104 KB
GPIO	35
PWM	14
Analog In	10
UART	2
SPI	Yes
I2C	Yes
Networking	Ethernet TCP/IP
CAN	Yes
Analog Out	2
USB Client	Debugging Only
USB Host	Yes
1-Wire	Yes
Real Time Clock	Yes

Tabulka 1 – Parametry FEZ Cerbuino Net [GHI Electronics]	
In-Field Update	No
Native LCD Controller	No
Supported Image Type	BMP
Memory Cards	Yes
File Systém	FAT16/FAT32
Operating Temp.	0 to +70°C
Dimensions	55.9 x 81.3 x 19.6 mm
Weight	28 g
RLP	Yes
RLP RAM Size	4KB
Gadgeteering Socket Count	3
Gadgeteering Socket Type	Y (x2) A (x2) I K O P (x2) S U (x2) X (x1)

5 Demonstrační úlohy

V rámci práce jsem se zabýval vypracováním demonstračních úloh, jakožto součásti vytváření studijních opor. Následující demonstrační úlohy jsou zaměřeny desku FEZ Cerbuino Net a STM32 F429 DISCOVERY, zde bylo nutné nejprve nahrát .NET Micro Framework, tento proces bude popsán níže, společně s patřičným připojením desek a nahráním projektů. Popis úloh je zároveň k dispozici v přílohách jako součást studijních opor a vychází z hlavní části textu, ale popisuje úlohy o něco podrobněji.

5.1 Umožnění komunikace desek s počítačem a založení projektu

Nejprve bude popsán proces zapojení pro jednotlivé desky, následně budou vysvětleny základy založení projektu a poté několik zpracovaných demonstračních úloh. Součást hlavního textu je popis, jakým bylo zapojení provedeno. Podrobnější návod zejména pro STM32 F429 DISCOVERY je součást studijních materiálů v přílohách a mimo jiné se zabývá i nutnými změnami nastavení systému Windows v případě používání novějších verzí jako Windows 8 a Windows 10.

5.1.1 Zapojení desky FEZ Cerbuino Net

Pro zapojení je potřeba mikro USB konektor, který slouží k napájení i komunikaci, ze stránek GHI Electronics jsou potřeba soubory (GHI Electronics NETMF SDK – aktuální verzi), které nám umožní konfiguraci komunikace a rovněž konfiguraci firmware desky na patřičnou verzi, dále je třeba stáhnout Microsoft Visual Studio a rozšíření Microsoft .NET Micro Framework 4.3, všechny verze se liší podle využívané desky a je potřeba se informovat na stránkách výrobce ohledně detailů. Při tomto procesu se objevil problém s instalací patřičné verze Microsoft .NET Micro Framework 4.3 do nejnovější verze Visual Studia 2015 a tím došlo k neúspěšným pokusům o vývoj a nahrání aplikace. Problém byl vyřešen instalací starší verze Visual Studia, konkrétně Visual Studia 2013, doporučené výrobcem desky. Po tomto se úspěšně zdařilo vytvořit aplikace, jež šli poté nahrát na desku.

5.1.2 Zapojení desky STM32 F429 DISCOVERY

Pro vývoj aplikací v .NET Micro Framework (NETMF) je zapotřebí několikero software. Některý může být méně běžný a proto je zapotřebí jej nejprve nainstalovat, než bude možné zahájit vývoj aplikací. Nejprve je nezbytné opět nainstalovat Microsoft Visual Studio, pro tuto desku je výrobcem doporučována verze Visual Studio 2012. Dále je zapotřebí nainstalovat Microsoft .NET Micro Framework SDK 4.3, poté je zapotřebí stáhnout balíček STM32F429I_Discovery_NETMF_Package: obsahující binární soubory, NETMF USB řadič, knihovny a příklady, vše potřebné pro vývoj a spuštění vlastních aplikací v .NETMF na desce STM32F429 Discovery. Rovněž je potřeba stáhnout a nainstalovat STM32 ST-LINK Utility: tento program se využívá k nahrání binárního kódu NETMF na desku STM32F429 Discovery. ST-LINK Utility automaticky nainstaluje ST-LINK/V2 USB řadič. STM32F429I_Discovery_NETMF_Package\NETMF_Binary_Image obsahuje NETMF binární image, který se skládá ze tří .hex souborů, tyto by měly být nahrány za využití STLINK Utility na desku STM32F429 Discovery. Rovněž je pro nahrání aplikací nezbytný STM32F429I_DISCOVERY NETMF USB řadič: USB USER (CN6), tento se využívá k vývoji a debugu NETMF aplikací na desce STM32F429 Discovery. Pro instalaci jsou zapotřebí následující kroky:

a) Zapojení STM32F429 Discovery do počítače za využití USB USER konektoru.

Systém poté detekuje zařízení "STM32F429 DISCOVERY".

b) Systém Windows poté potřebuje specifikaci ovladače, pro ten musíme navigovat do složky STM32F429I_Discovery_NETMF_Package a do podložky s USB řadičem, který je stažen jako součást balíčku.

c) Po dokončení instalace by se měl tento ovladač objevit ve správci zařízení: STM32F429I_DISCOVERY NETMF.

Při instalaci ovladače se objevily potíže s novějšími verzemi systému Windows, které neumožňují instalaci ovladačů bez digitálního podpisu, tento problém se musí vyřešit změnou nastavení systému, přesný návod se nachází na stránkách podpory Microsoft.

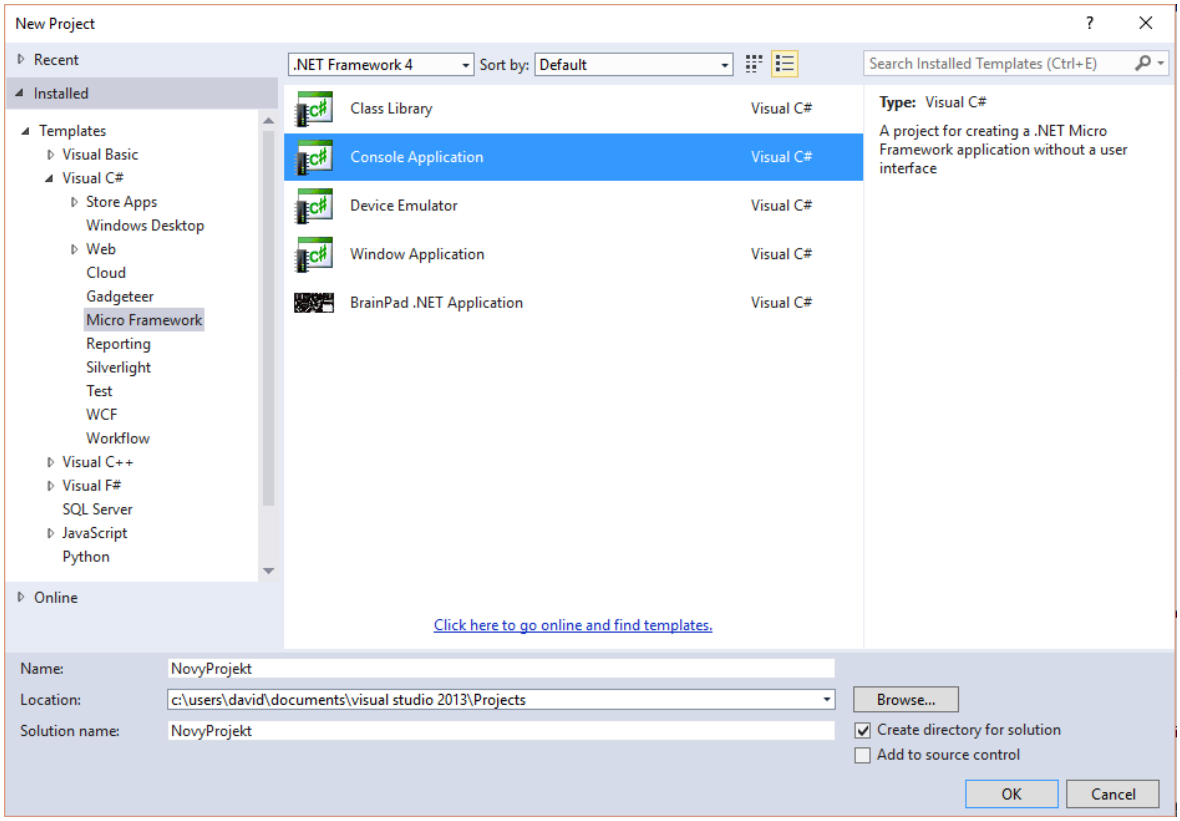
5.1.3 Založení projektu

Krátké naznačení založení projektu. V Microsoft Visual Studiu, pro některé desky existuji dvě možné alternativy založení projektu. První z těchto možností je založit Console Application přes Micro Framework záložku, druhou možností je založit přes záložku Gadgeteer aplikaci pro Gadgeteery, při využití této varianty je Visual Studiem vytvořen kód, který se specificky přizpůsobí zapojené desce, práce tímto způsobem je snazší, avšak o něco méně flexibilní. Následující kroky a obrázek jasně popisují založení nové projektu krok po kroku.

1. Spuštění Microsoft Visual Studia.

2. V hlavní nabídce vybrat File > New Project

3. Podle popisu výše zvolit jednu z alternativ a vytvořit nový projekt.



Obrázek 3 – Založení projektu (Visual Studio)

Dále je na řadě vytvoření samotného programu, příklady různých projektů jsou níže s příklady kódů a základními popisy funkce. Příklady jsou vypracovány pro obě alternativy, jelikož pro desku FEZ Cerbuino Net se prokázalo jako výrazně snazší vytvořit Gadgeteer

aplikace, zatímco pro desku STM32 F429 DISCOVERY není varianta pro vytvoření Gadgeteer aplikace a využívá se tedy alternativy Konzolové aplikace pro vytvoření projektů.

5.2 FEZ Cerbuino Net – základní úloha, blikající dioda

Jako první úloha pro desku FEZ Cerbuino Net byla pouze blikající dioda, za cíl bylo zejména seznámení se s funkcí desky, programovacím prostředím a rovněž jakým stylem jsou programy psány. Funkcí aplikace je rozblikání diody přítomné na desce, toto blikání je zcela automatické s časovým intervalem 700 ms a spouští se okamžitě po nahrání programu na desku. Pro programovaní bylo zvoleno Visual Studio a programovací jazyk C#. Program je sestaven jako aplikace Gadgeteer, kvůli jednoduchosti provedení a snadné možnosti zkontrolovat správnost kódu a dohledat případné řešení chyb a to díky snadné přístupnosti a jednoduchého dohledání řady návodů.

```
namespace HelloLED
{
    public partial class Program
    {
        void ProgramStarted()
        {
            GT.Timer timer = new GT.Timer(700);
            timer.Tick += timer_Tick;
            timer.Start();
        }
        bool state = false;
        void timer_Tick(GT.Timer timer)
        {
            state = !state;
            Mainboard.SetDebugLED(state);
        }
    }
}
```

Obrázek 4 – Hlavní část programu pro FEZ Cerbuino Net Dioda (Visual Studio)

5.3 FEZ Cerbuino Net – jednoduché nastavení GPIO

Další úlohou pro tuto desku je zapojení a spínání diody v závislosti na vstupu. I zde je pro jednoduchost aplikace vytvářena přes Gadgeteer. Jsou zavedeny 2 výstupní porty, z nichž jeden je neustále sepnutý, aby modul fungoval, druhý náhodně zvolený pin se rozsvěcuje a zhasíná v závislosti, jestli je detekována 1 – nesvítí, nebo 0 – svítí. Blikající dioda je na rozdíl od předchozího příkladu přítomná na rozšiřujícím modulu, ale obdobně blikání je zcela

automatické, nyní s intervalem 500 ms, spouští se rovněž po nahrání programu na desku. V úloze šlo zejména o seznámení se s adresováním a referencemi. Při řešení úlohy nastala komplikace, protože poskytnuté materiály popisovaly práci se starším hardwarem a popis pro SDK 4.1, zatímco pro práci byl zvolen SDK 4.3 a to z důvodů, že byl nejnovější verze a navíc doporučen výrobcem hardware jako optimální pro řešení úloh. To vedlo ke komplikacím právě v oblasti referencí a tím zároveň adresování, program měl problém rozpoznat a reagovat na příkazy. Zároveň byla odlišnost oproti materiálům ve faktu, že v nich byly programy popsány přímo pro .NET Micro Framework a nikoli pro Gadgeteer aplikace, což vedlo k odlišnostem v kódu. Tyto problémy byly vyřešeny ve spolupráci s vedoucím práce.

```
namespace gpio3
{
    public partial class Program
    {
        OutputPort led;
        OutputPort out13;
        bool ledState = false;
        void ProgramStarted()
        {
            led = new OutputPort((Cpu.Pin)GHI.Pins.FEZCerbuinoNet.Headers.Gpio.D8, ledState);
            out13 = new OutputPort((Cpu.Pin)GHI.Pins.FEZCerbuinoNet.Headers.Gpio.D13, true);
            GT.Timer timer = new GT.Timer(500);
            timer.Tick += timer_Tick;
            timer.Start();
            Debug.Print("Program Started");
        }

        void timer_Tick(GT.Timer timer)
        {
            ledState = !ledState;
            led.Write(ledState);
        }
    }
}
```

Obrázek 5 – Hlavní část programu pro FEZ Cerbuino Net GPIO (Visual Studio)

5.4 FEZ Cerbuino Net – Ethernet

Třetím programem je nastavení komunikace Ethernet, tento program je z velké části zpracován ve spolupráci s vedoucím práce a má za cíl nejen aktivaci samotného Ethernet portu na desce, ale zároveň i jeho využití. Za pomocí kódu byla na desce umožněna komunikace prostřednictvím LAN kabelu a poté je na desce vytvořen server pro komunikaci s počítačem, program obsahuje jeho inicializaci, proces a ukončení, zároveň obsahuje komunikační kód. Rovněž je k dispozici aplikace umožňující počítači komunikovat s takto vytvořeným serverem. S přihlédnutím k zadání kladu hlavní důraz na umožnění samotné

komunikace Ethernetem, kterou tato poskytnutá deska podporuje. Na obrázku níže můžeme vidět část kódu, která mimo jiné inicializuje Ethernet na desce, využitý postup je rovněž k nalezení na oficiálních stránkách GHI Electronics. I při umožňování komunikace podle návodu výrobce nastaly drobné problémy, vzhledem k tomu, že byla zvolena Gadgeteer aplikace z důvodu, že psaní kódu je snazší i díky strukturování zvolenému Visual Studiem. Tyto problémy byly opět vyřešeny za asistence vedoucího práce. Další část kódu je značně rozsáhlá a obsahuje několik částí plnící funkce umožňující komunikaci s aplikací na straně počítače (jedná se o inicializace serverů, generování dat a různá ošetření). Na obrázky můžeme také vidět kód zakládající server.

```
static NetworkInterface iface;
const Int32 Port = 2112;
IPEndPoint BindingEndPoint;
Socket Server = null;
Socket Client = null;
Thread ServerProcess = null;
Thread ClientCommunicationProcess = null;
GT.Timer timerServerStarted;        //Indikátor funkčního založení serveru
GT.Timer timerClientConnected;    //Indikátor navázání spojení, připojení klienta
int len;
bool LedOnOff = true;
// This method is run when the mainboard is powered up or reset.
void ProgramStarted()
{
    NetworkChange.NetworkAvailabilityChanged += new NetworkAvailabilityChangedEventHandler(NetworkChange_NetworkAvailabilityChanged);
    iface = NetworkInterface.GetAllNetworkInterfaces()[0] ;
    //iface.EnableStaticIP("192.168.0.101", "255.255.255.0", "");
    iface.EnableStaticIP("192.168.5.200", "255.255.255.0", "");

    timerServerStarted = new GT.Timer(1000); // every second (1000ms)
    timerServerStarted.Tick += new GT.Timer.TickEventHandler(timer_Tick);
    timerClientConnected = new GT.Timer(250); // every half second (500ms)
    timerClientConnected.Tick += new GT.Timer.TickEventHandler(timer_Tick);

    ServerStart(); //Pokus o založení serveru

    relay.SetRelay(Relay.AvailableRelays.Relay4, true);
    relay.SetRelay(Relay.AvailableRelays.Relay3, true);
    relay.SetRelay(Relay.AvailableRelays.Relay2, true);
    relay.SetRelay(Relay.AvailableRelays.Relay1, false);

    Mainboard.SetDebugLED(true); //Proces končí rozsvícením diody (je-li úspěšně server úspěšně založen, bude dioda blikat)
}

//Událost blikání diody
void timer_Tick(GT.Timer timer)
{
    Mainboard.SetDebugLED(LedOnOff = !LedOnOff);
}

//Událost fyzické změny stavu Sítě (Fyzická vrstva je/není k dispozici)
void NetworkChange_NetworkAvailabilityChanged(object sender, NetworkAvailabilityEventArgs e)
{
    if (e.IsAvailable)
    {
        Debug.Print("!! Network Available...");
        ServerStart();
    }
    else
    {
        Debug.Print("!! Network Unavailable...");
        ServerStop();
    }
}
```

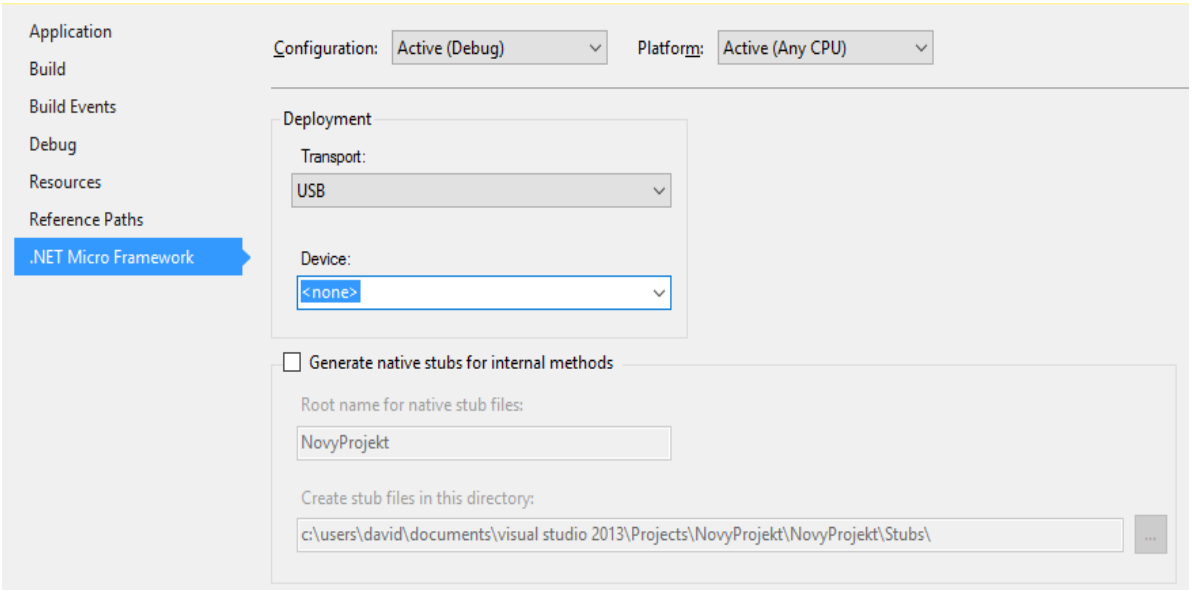
Obrázek 6 – Hlavní část programu pro FEZ Cerbuino Net Ethernet (Visual Studio)

5.5 STM32 F429 DISCOVERY – GPIO

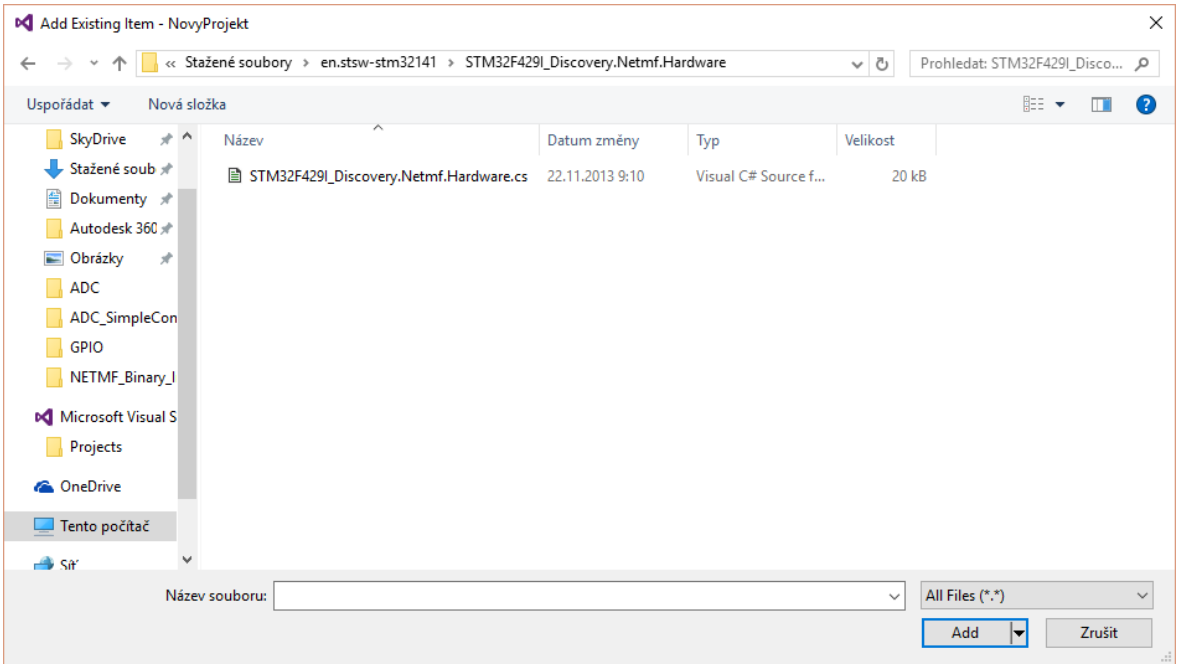
Jako první úloha pro desku STM32 F429 DISCOVERY byla sestavena úloha s blikající LED, za cíl bylo zejména seznámení se s funkcí desky, programovacím prostředím a rovněž jakým stylem jsou programy psány pro tuto desku obzvláště v této situaci, kdy byl .NET Micro Framework nahrán ručně a není na desce od začátku. Úloha spočívá v inicializaci LED diody a poté nastavení cyklu, v jakém bude daná dioda přepínat mezi červenou a zelenou. Blikání je automatické v intervalu 750 ms pro červenou a poté stejně dlouho pro zelenou a je inicializováno okamžitě po nahrání programu. Pro programovaní bylo opět zvoleno Visual Studio a programovací jazyk C#. Program byl vytvořen jako konzolová aplikace a vychází z materiálů poskytnutých společností STMicroelectronics. Samotný program je jednoduchý a v případě následování výše zmíněných instrukcí by mělo sestavení programů být snadné. Pro nahrání programu na desku je ještě nezbytné upravit nastavení ve Visual Studiu. Je třeba zvolit záložku Project, v bočním menu zvolit .NET Micro Framework a pod sekcí Deployment zvolit pro Transport možnost USB. Pokud jsme při konfiguraci USB komunikace pro desku postupovali správně, mělo by se pro Device automaticky zobrazit připojené zařízení. Pro správnou adresaci je rovněž nutné přidat STM32F429I_Discovery.Netmf.Hardware. Přidání se provede pravým kliknutím na název projektu v Solution Explorer, zvolit Add a možnost Existing Item a zvolit složku s takto pojmenovaným souborem.

```
using System.Threading;
using Microsoft.SPOT.Hardware;
using STM32F429I_Discovery.Netmf.Hardware;


public class BlinkingLed
{
    public static void Main()
    {
        LED.LEDInit(); // Inicializace LED GPIO

        while (true)
        {

            LED.RedLedToggle();
            Thread.Sleep(750);
            LED.GreenLedToggle();
            Thread.Sleep(750);

        }
    }
}
```

Obrázek 7 – Hlavní část programu pro STM32 F429 DISCOVERY GPIO (Visual Studio)



Obrázek 8 – Nastavení nahrání projektu na desku (Visual Studio)



Obrázek 9 – Lokalizace STM32F429I_Discovery.Netmf.Hardware (Visual Studio)

5.6 STM32 F429 DISCOVERY – ADC

Dalším programem je A/D převodník, který je rozsahově jednoznačně větším projektem, kdy program nejprve načte analogové vstupy a poté přečte hodnoty na těchto vstupech, podle možností desky můžeme naprogramovat několik vstupů, jež budou čteny programem. Následně program vypíše hodnoty přečtené na vstupech do okna Output v programu Visual Studio, zároveň je opět inicializována LED, jež se zapne během cyklu, po jeho proběhnutí je zapnuto vyčkávání na 2 sekundy, Funkce LED je zde jako indikátor probíhající aplikace na desce, proto je zvolena zelená barva. Program je rovněž vypracován na základě materiálů od společnosti STMicroelectronics. Pro jeho nahrání na desku je opět třeba následovat stejného postupu jako u minulého projektu s nastavením komunikace přes USB ve vlastnostech projektu a opět nahrát STM32F429I_Discovery.Netmf.Hardware.

```
using System;
using Microsoft.SPOT;
using System.Threading;
using Microsoft.SPOT.Hardware;
using STM32F429I_Discovery.Netmf.Hardware;

namespace ADKonv
{
    public class Program
    {
        public static void Main()
        {
            AnalogInput ADC0 = new AnalogInput(ADC.Channel0_PA6);

            AnalogInput ADC1 = new AnalogInput(ADC.Channel1_PA7);

            AnalogInput ADC2 = new AnalogInput(ADC.Channel2_PC1);

            AnalogInput ADC3 = new AnalogInput(ADC.Channel3_PC3);

            /* Inicializace LED */
            LED.LEDInit();

            while (true)
            {
                /* Zobrazení ADC převedené hodnoty */
                Debug.Print("Channel0 (pin " + ADC0.Pin + ") = " + (ADC0.Read() * 3.3).ToString("f2") + "V");
                Debug.Print("Channel1 (pin " + ADC1.Pin + ") = " + (ADC1.Read() * 3.3).ToString("f2") + "V");
                Debug.Print("Channel2 (pin " + ADC2.Pin + ") = " + (ADC2.Read() * 3.3).ToString("f2") + "V");
                Debug.Print("Channel3 (pin " + ADC3.Pin + ") = " + (ADC3.Read() * 3.3).ToString("f2") + "V");


                Thread.Sleep(2000);

                LED.GreenLedToggle();
            }
        }
    }
}
```

Obrázek 10 – Hlavní část programu pro STM32 F429 DISCOVERY ADC (Visual Studio)

6 Závěr

.NET Micro Framework představuje velice praktický nástroj při tvorbě vlastních jednoduchých elektronických zařízení. Jeho přínos spočívá v jednoduchosti a flexibilitě, kdy je zde možno na obrovském množství hardwaru a softwaru aplikovat nejrůznější předchozí zkušenosti. Dalším přínosem jeho jednoduchosti je možnost využití pro naprosté začátečníky, aby se seznámili s elektronikou a programováním. Obzvláště přínosný je fakt, že s výjimkou hardwaru, není nutná žádná jiná investice.

V této práci jsem si kladl zejména za cíl seznámení s teoretickou částí .NET Micro Framework, jeho funkcemi, možnostmi a dostupností. Dále jsem se zde zabýval hardwarem, jež je u nás dostupný a jeho možnostmi, s bližším zaměřením na poskytnuté pomůcky k této práci. V rámci práce jsem se zabýval vytvářením demonstračních programů, pro ukázku možností platformy.

Během práce s touto platformou jsem narazil na několik potíží a to zejména nedostatek podpory při práci se softwarem, jako byla v mém případě při komplikace s Microdoft Visual Studiem 2015.

Další z problémů je nedostatkem návodů, které obvykle pokrývají jen základní funkce a nastavení desek. Zde opět narážíme na softwarovou bariéru. Dostupné materiály byly značně rozsáhlé a pokrývaly dost oblastí, ale situace je komplikována tím, že jsou do značné míry specifické v tehdejší době aktuálnímu softwaru a hardwaru, který od té doby prošel značným vývojem.

Při zpracování práce jsem dospěl k závěru, že technologie .NET Micro Framework má potenciál mít značný přínos v oblasti mikroelektroniky nejen ve volnočasové aktivitě, ale i profesionální sféře a to i přes výše zmíněné potíže. Vývoj tímto směrem je však brzděn značně omezenou podporou ze strany vývojářů, kteří převážně pouze organizují výzkum prováděny nadšenou komunitou.

To znamená, že chce-li se někdo začít zabývat vývojem vlastních aplikací, potřebuje se věnovat procházení řady fór a zároveň věnovat spoustu času samostatnému experimentování a pokusům o vývoj aplikací. Obrovskou pomocí je zároveň praxe s programovacím jazykem využívaným .NET Micro Framework i mimo programování aplikací pouze pro něj.

Pro snazší úvod do problematiky bylo jednou z náplní práce zpracování částí textu do podoby studijních materiálů, jež mohou sloužit jako východisko pro seznámení se s platformou .NET Micro Framework a s programováním základních funkcí. Tyto materiály jsou součást příloh.

Poděkování

Touto formou bych chtěl poděkovat Ing. Davidu Fojtíkovi, Ph.D. za poskytnutí nezbytných pomůcek a materiálů nezbytných k vypracování práce, jakož i poskytnutí konzultací nezbytných k dokončení programů. Dále bych chtěl poděkovat doc. Ing. Renatě Wagnerové, Ph.D. za poskytnutí konzultací ohledně formy, jakou budou provedeno vypracování.

7 Použitá literatura

.NET Micro Framework [online]. Dostupný z <URL:https://netmf.github.io/>

GALLI P. 2009. *Microsoft to Open Source the .NET Micro Framework* [online]. Dostupný z <URL:http://blogs.technet.com/b/port25/archive/2009/11/16/microsoft-to-open-source-the-net-1micro-framework.aspx>

VBTEAM 2011. *Micro-framework v4.2 Support for Visual Basic.* [online]. Dostupný z <URL:http://blogs.msdn.com/b/vbteam/archive/2011/06/08/micro-framework-v4-2-support-for-visual-basic.aspx>

STMICROELECTRONICS 2014. *STM32F4DISCOVERY*. [online]. Dostupný z <URL:http://www.st.com/st-web-ui/static/active/en/resource/technical/document/data_brief/DM00037955.pdf>

GHI ELECTRONICS *FEZ Cerbuino Net.* [online]. Dostupné z <URL:https://www.ghielectronics.com/catalog/product/473>

MICROSOFT 2015. *.NET Gadgeteer – Microsoft Research* [online]. Dostupné z <URL:http://research.microsoft.com/en-us/projects/gadgeteer/>

MICROSOFT OPEN TECHNOLOGIES 2014. *Gadgeteer| .NET Micro Framework* [online]. Dostupné z <URL:http://www.NETmf.com/gadgeteer/>

.NET Micro Framework e-shop [online]. Dostupné z <URL: http://shop.microframework.eu/Home.aspx >

8 Seznam příloh

Studijní opory

CD s prací

VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA

FAKULTA STROJNÍ

Katedra automatizační techniky a řízení

Studijní opory pro .NET Micro Framework

Obsah

1 ZAPOJENÍ HARDWARU

Po úspěšném a aktivním absolvování této KAPITOLY

Budete umět:	Budete umět
Zapojit desku FEZ Cerbuino Net, tak aby bylo možné zahájit programování Zapojit desku STM32 F429 DISCOVERY a nahrát na ni .NET Micro Framework	

Budete schopni:	Budete schopni
Zahájit programování svých aplikací na zmíněné desky	

.NET Micro Framework je vývojové a realizační prostředí pro zařízení, jež operuje s omezenými zdroji, tato kapitola popíše, jakým způsobem zapojit desky, aby na ně bylo možno nahrát aplikace.

1.1 Zapojení FEZ Cerbuino Net

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

Zapojit FEZ Cerbuino Net
Provést Update firmware
Připravit desku k vývoji aplikací

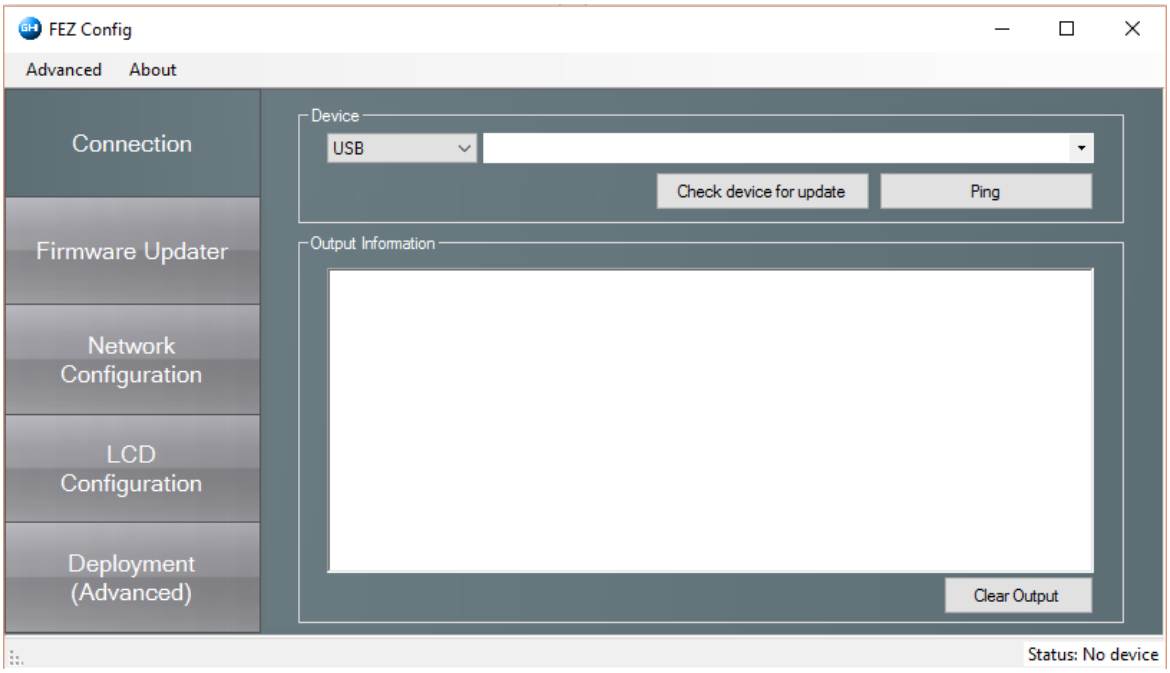
Výklad

FEZ Cerbuino Net je Arduino-kompatibilní .NET Gadgeteer základní deska, na které běží .NET Micro Framework, což umožňuje uživatelům programovat desky z Visual Studio Microsoftu za využití jazyků C# nebo Visual Basic, což umožňuje vývojářům využít výhod rozsáhlých zabudovaných knihoven pro sítě, souborové systémy, grafické rozhraní a periferní zařízení.

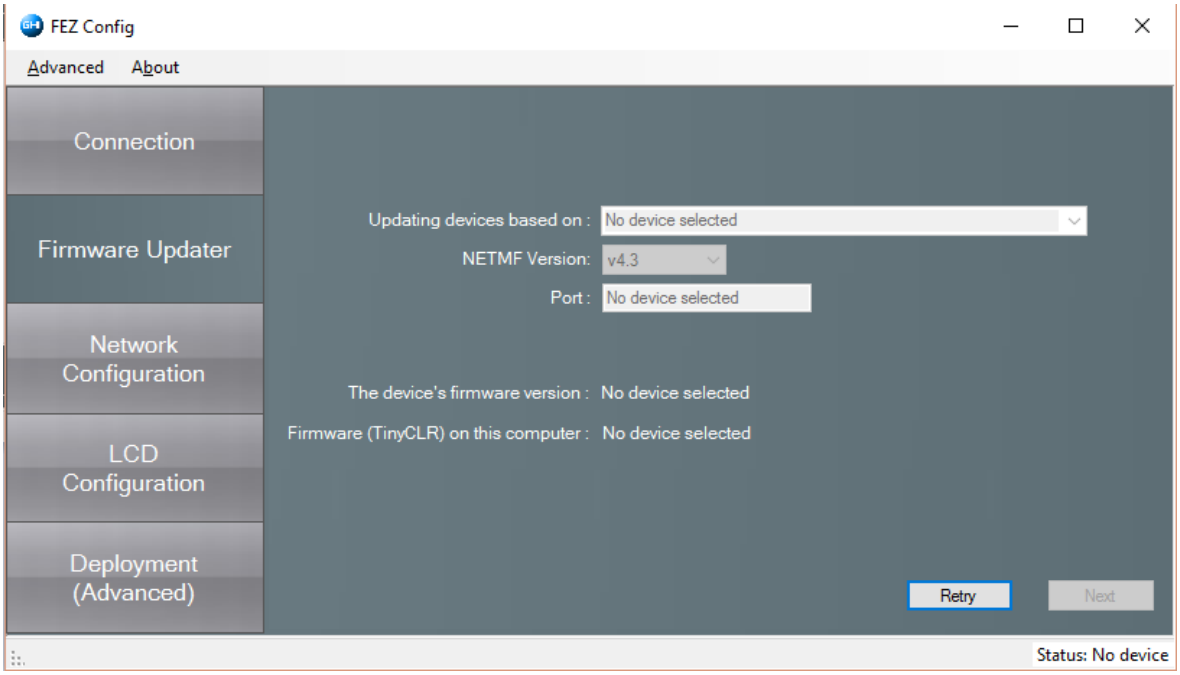
Zapojení desky

Desku se zapojí pomocí USB konektoru, v případě této desky nedochází k žádným komplikacím, jelikož na ní již ze startu běží .NET Micro Framework, pro správnou komunikaci je třeba ovšem provést update firmware prostřednictví softwaru FEZ Config, jež je snadno dostupný na stránkách GHI Electronics. Dále už je jen potřeba stáhnout Microsoft Visual Studio, kdy výrobce doporučuje verzi VS 2013, ale v současné době by neměl být problém ani s verzí VS 2015. Dále je potřeba stáhnout Microsoft .NET

Micro Framework 4.3 (v případě jiné desky použít verzi upřesněnou výrobcem) a GHI Electronics NETMF SDK v patřičné verzi.



Obr. 1. FEZ Config



Obr. 2 Firmware Updater

Shrnutí kapitoly

Kapitola popisuje jak zapojit desku FEZ Cerbuino Net tak, aby bylo možné s ní dále pracovat a vyvíjet pro ni aplikace.

Kontrolní otázka

Kapitola nevyžaduje kontrolní otázky.

Úkol k řešení

1. Stáhnout potřebný software

Vyzkoušet konfiguraci, aby počítač byl schopen komunikace s deskou

1.2 Zapojení STM32 F429 DISCOVERY

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

Zapojit STM32 F429 DISCOVERY
Nahrát .NET Micro Framework na desku
Připravit desku k vývoji aplikací

Výklad

STM32 F4 DISCOVERY přináší řadu příslušenství a možnost snadno vyvinout vlastní aplikace. Zahrnuje vše potřebné pro začátečníky i zkušené uživatele, aby mohli rychle začít.

Zapojení desky

Pro vývoj aplikací v .NET Micro Framework (NETMF) je zapotřebí několikero software. Některý může být méně běžný a proto je zapotřebí jej nejprve nainstalovat, než bude možné zahájit vývoj aplikací. Nejprve je nezbytné opět nainstalovat Microsoft Visual Studio, pro tuto desku je výrobcem doporučována verze Visual Studio 2012. Dále je zapotřebí nainstalovat Microsoft .NET Micro Framework SDK 4.3, poté je zapotřebí stáhnout balíček STM32F429I_Discovery_NETMF_Package: obsahující binární soubory, NETMF USB řadič, knihovny a příklady, vše potřebné pro vývoj a spuštění

vlastních aplikací v .NETMF na desce STM32F429 Discovery. Rovněž je potřeba stáhnout a nainstalovat STM32 ST-LINK Utility: tento program se využívá k nahrání binárního kódu NETMF na desku STM32F429 Discovery. ST-LINK Utility automaticky nainstaluje ST-LINK/V2 USB řadič. STM32F429I_Discovery_NETMF_Package\NETMF_Binary_Image obsahuje NETMF binární image, který se skládá ze tří .hex souborů, tyto by měly být nahrány za využití STLINK Utility na desku STM32F429 Discovery. Rovněž je pro nahrání aplikací nezbytný STM32F429I_DISCOVERY NETMF USB řadič: USB USER (CN6), tento se využívá k vývoji a debugu NETMF aplikací na desce STM32F429 Discovery. Pro instalaci jsou zapotřebí následující kroky:

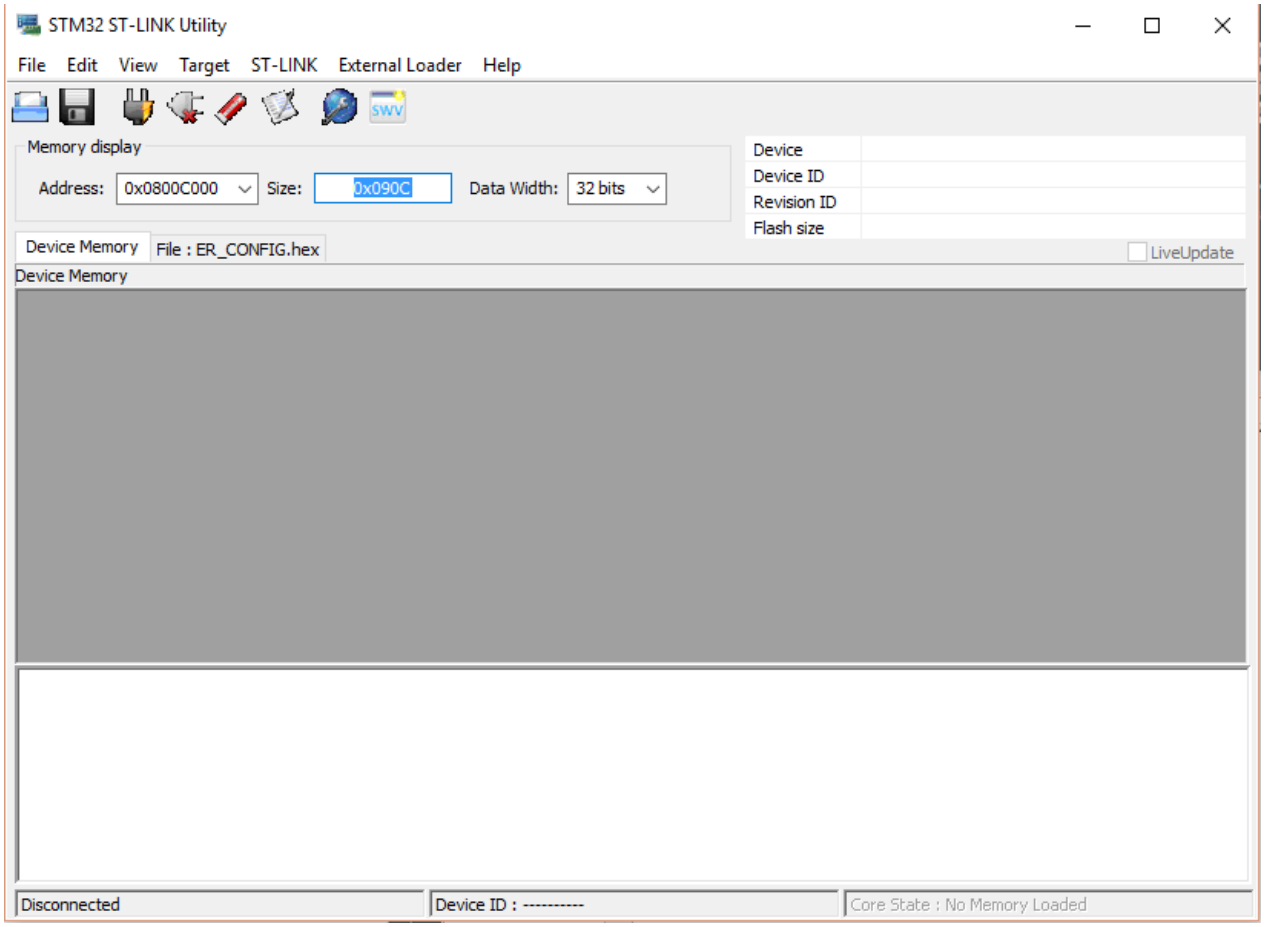
a) Zapojení STM32F429 Discovery do počítače za využití USB USER konektoru.

Sestém poté detekuje zařízení "STM32F429 DISCOVERY".

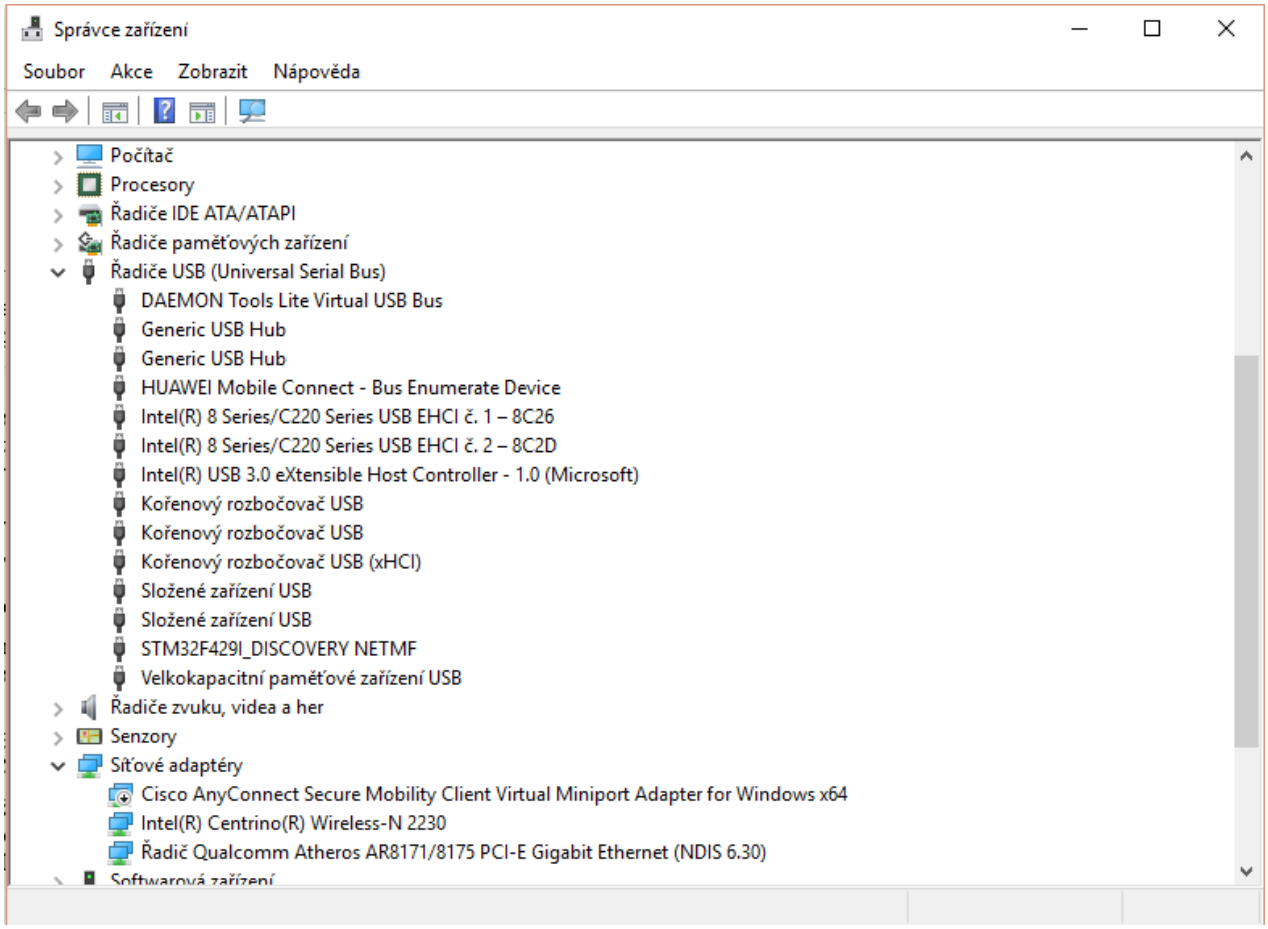
b) Systém Windows poté potřebuje specifikaci ovladače, pro ten musíme navigovat do složky STM32F429I_Discovery_NETMF_Package a do podložky s USB řadičem, který je stažen jako součást balíčku.

c) Po dokončení instalace by se měl tento ovladač objevit ve správci zařízení: STM32F429I_DISCOVERY NETMF.

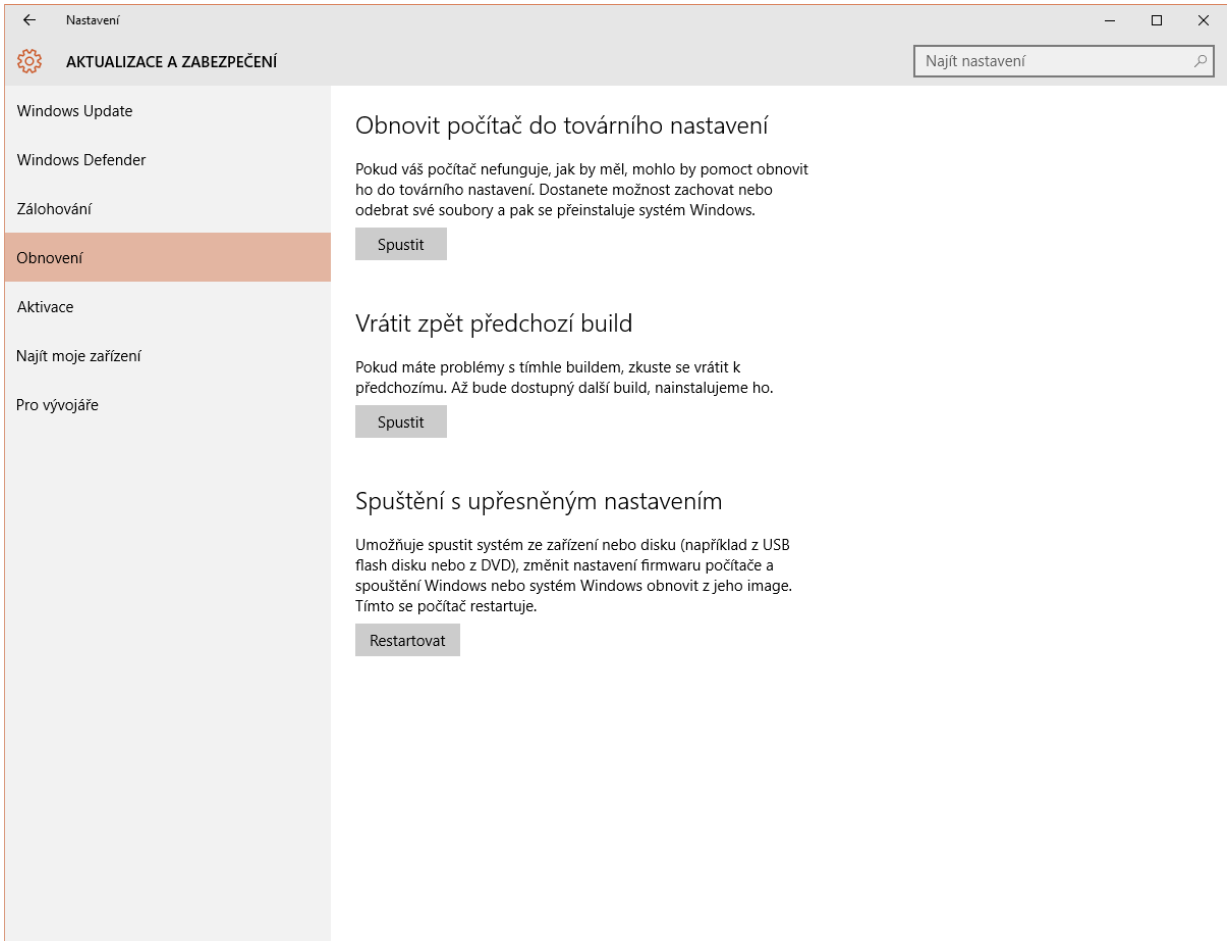
Při instalaci ovladače se objevily potíže s novějšími verzemi systému Windows, které neumožňují instalaci ovladačů bez digitálního podpisu, tento problém se musí vyřešit změnou nastavení systému, přesný návod se nachází na stránkách podpory Microsoft.



Obr. 3. STM32 ST-LINK Utility



Obr. 4 Správce zařízení



Obr. 5 Navigace pro změnu nastavení ve Windows – zvolit Restartovat
(další kroky doporučeno následovat podle oficiálního návodu)

Shrnutí kapitoly

Kapitola popisuje zapojení STM32 F4 DISCOVERY, nahrání .NET Micro Framework a umožnění komunikace s počítačem tak, aby bylo možno nahrát .NET Micro Framework aplikace.

Kontrolní otázka

Kapitola nevyžaduje kontrolní otázky.

Úkol k řešení

1. Stáhnout potřebný software
2. Vyzkoušet konfiguraci, aby počítač byl schopen komunikace s deskou

2 ZALOŽENÍ NOVÉHO PROJEKTU

Po úspěšném a aktivním absolvování této KAPITOLY

Budete umět:	
Založit nový projekt ve Visual Studiu pro .NET Micro Framewrok	Budete umět

.NET Micro Framework je vývojové a realizační prostředí pro zařízení, jež operuje s omezenými zdroji, tato kapitola popíše, jakým způsobem zapojit desky, aby na ně bylo možno nahrát aplikace.

2.1 Založení projektu ve Visual Studiu

Čas ke studiu: 30 minut

Cíl Po prostudování tohoto odstavce budete umět

Zapojit FEZ Cerbuino Net
Provést Update firmware
Připravit desku k vývoji aplikací

Výklad

Projekt lze vytvořit jako aplikace pro Gadgeteer, cože je varianta jednodušší, ale potenciálně méně praktická, nebo jak Konzolová aplikace. Obě alternativy jsou popsány níže.

Zapojení desky

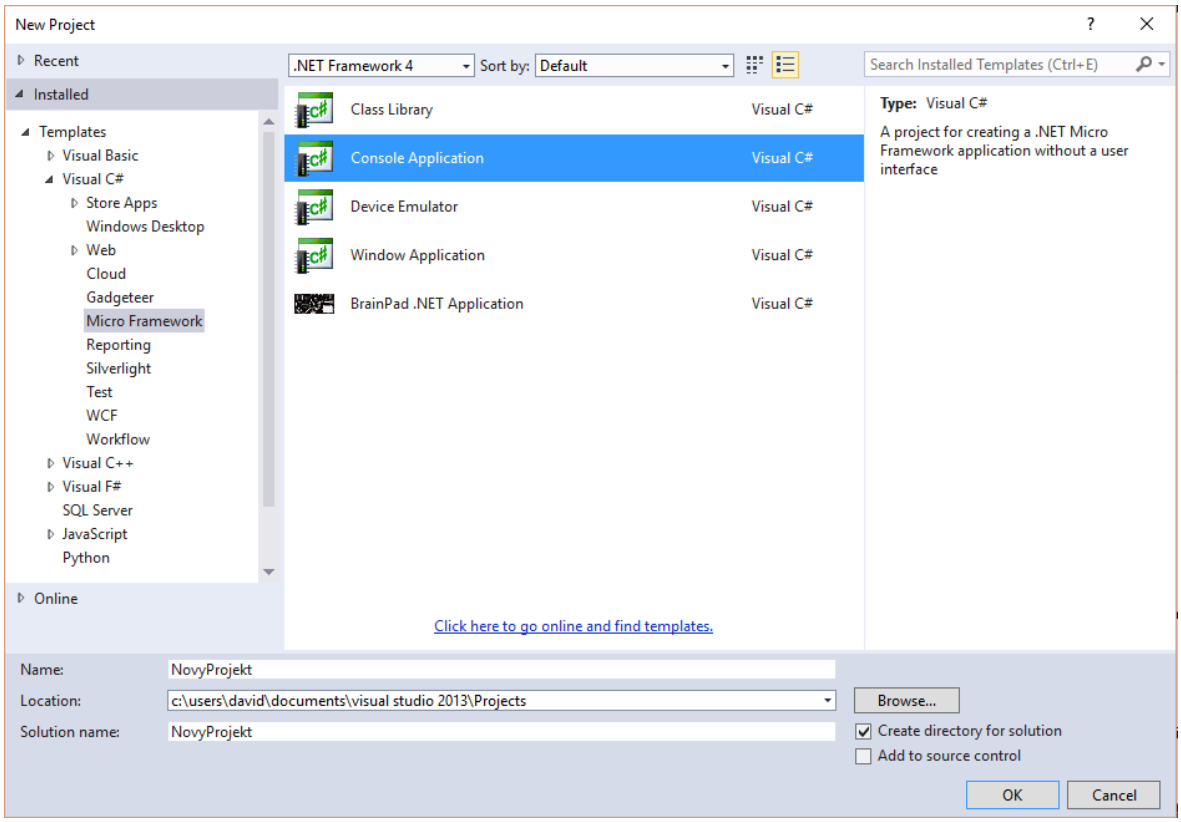
V Microsoft Visual Studiu, pro některé desky existuji dvě možné alternativy založení projektu. První z těchto možností je založit Console Application přes Micro Framework záložku, druhou možností je založit přes záložku Gadgeteer aplikaci pro Gadgeteery, při využití této varianty je Visual Studiem vytvořen kód, který se specificky přizpůsobí zapojené desce, práce tímto způsobem je snazší, avšak o něco méně flexibilní. Následující kroky a obrázek jasně popisují založení nové projektu krok po kroku.

1. Spuštění Microsoft Visual Studia.

2. V hlavní nabídce vybrat File > New Project

3. Podle popisu výše zvolit jednu z alternativ a vytvořit nový projekt

Dále je na řadě vytvoření samotného programu, příklady různých projektů jsou níže s příklady kódů a základními popisy funkce. Příklady jsou vypracovány pro obě alternativy, jelikož pro desku FEZ Cerbuino Net se prokázalo jako výrazně snazší vytvořit Gadgeteer aplikace, zatímco pro desku STM32 F429 DISCOVERY není varianta pro vytvoření Gadgeteer aplikace a využívá se tedy alternativy Konzolové aplikace pro vytvoření projektů



Obr. 6. Založení projektu

Shrnutí kapitoly

Kapitola popisuje jak založit projekt v Microsoft Visual Studiu, popis je proveden pro Microsoft Visual Studio 2013.

Kontrolní otázka

1. Jaké jsou alternativy projektů pro desky.
2. Jaké jsou výhody a nevýhody jednotlivých alternativ.

Úkol k řešení

1. Vyzkoušet si založení projektu ve Visual Studiu

3 ÚLOHY PRO FEZ CERBUINO NET

Po úspěšném a aktivním absolvování této KAPITOLY

Budete umět:	Budete umět
Vytvořit několik aplikací pro FEZ Cerbino Net Nahrát vytvořené aplikace na desku	

Budete schopni:	Budete schopni
Vyzkoušet si spuštění aplikací na desce	

V následující kapitole jsou popsány tři úlohy na desku FEZ Cerbuino Net včetně kódů, které umožní naprogramování základních funkcí a mohou sloužit jako základ pro komplikovanější projekty.

3.1 FEZ Cerbuino Net – základní úloha, blikající dioda

Čas ke studiu: 30 minut

Cíl Po prostudování tohoto odstavce budete umět

Rozblikat diodu na desce podle zvolených parametrů

Výklad

Jako první úloha pro desku FEZ Cerbuino Net je pouze blikající dioda, za cíl je zejména seznámení se s funkcí desky, programovacím prostředím a rovněž jakým stylem jsou programy psány.

Program

Pro programovaní je použito Microsoft Visual Studio 2013 a programovací jazyk C#. Program je sestaven jako aplikace Gadgeteer, kvůli jednoduchosti provedení a snadné možnosti zkontrolovat správnost kódu a dohledat případné řešení chyb a to díky snadné přístupnosti a jednoduchému dohledání řady návodů. Vše potřebné bylo vysvětleno již výše a jediné, co je potřeba je vytvoření samotného kódu. Kód na obrázku níže je vše co je nezbytné sepsat, zbytek je vytvořen přímo Visual Studiem, jelikož program je na velice základní úrovni, není potřeba dalších úprav.

```
namespace HelloLED
{
    public partial class Program
    {
        void ProgramStarted()
        {
            GT.Timer timer = new GT.Timer(700);
            timer.Tick += timer_Tick;
            timer.Start();
        }
        bool state = false;
        void timer_Tick(GT.Timer timer)
        {
            state = !state;
            Mainboard.SetDebugLED(state);
        }
    }
}
```

Obr. 8. Hlavní část programu pro FEZ Cerbuino Net Dioda

3.2 FEZ Cerbuino Net – jednoduché nastavení GPIO

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

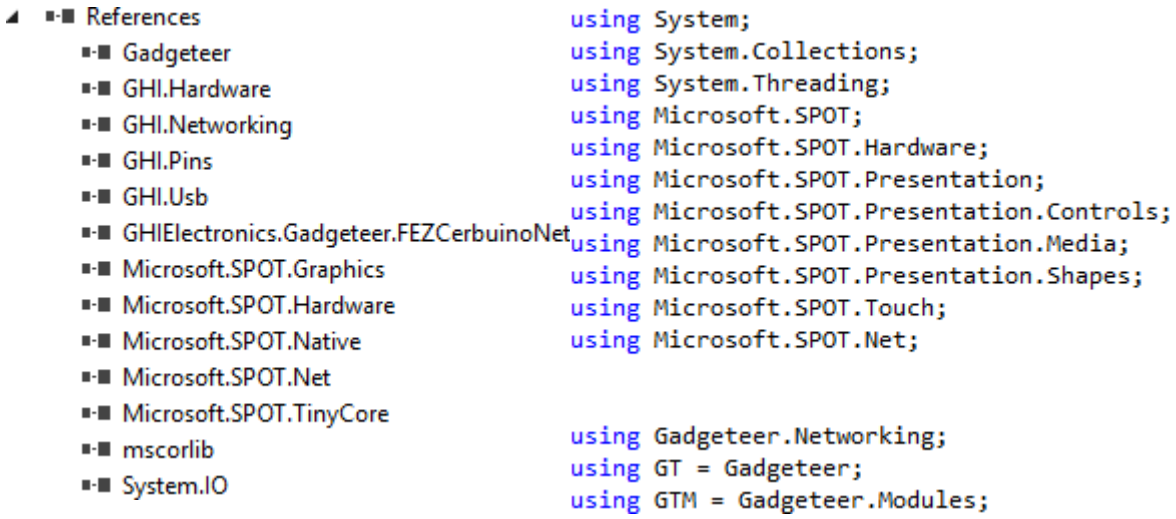
Nastavit GPIO pro desku FEZ Cerbuino Net
Rozblikat diody na připojeném modulu

Výklad

Další úlohou pro tuto desku je zapojení a spínání diody v závislosti na vstupu. I zde je pro jednoduchost aplikace vytvářena přes Gadgeteer.

Program

Jsou zavedeny 2 Outporty, z nichž jeden je neustále sepnutý, aby modul fungoval, druhý náhodně zvolený pin se rozsvěcuje a zhasíná v závislosti, jestli je detekována 1 – nesvítí, nebo 0 – svítí. V úloze se jedná zejména o seznámení se s adresováním a referencemi. Při tomto programu je tedy třeba manuálně přidat řadu referencí, které standardně v programu zahrnuty nejsou, doplníme je pomocí založení aplikace pro naši desku, což nám poté umožní přidat pod položkou reference vše potřebné. Přidání je provedeno pravým kliknutím na References a zvolením možnosti Add References.



Obr. 9. Přehled využitých referencí a úvodní část programu

```
namespace gpio3
{
    public partial class Program
    {
        OutputPort led;
        OutputPort out13;
        bool ledState = false;
        void ProgramStarted()
        {
            led = new OutputPort((Cpu.Pin)GHI.Pins.FEZCerbuinoNet.Headers.Gpio.D8, ledState);
            out13 = new OutputPort((Cpu.Pin)GHI.Pins.FEZCerbuinoNet.Headers.Gpio.D13, true);
            GT.Timer timer = new GT.Timer(500);
            timer.Tick += timer_Tick;
            timer.Start();
            Debug.Print("Program Started");
        }

        void timer_Tick(GT.Timer timer)
        {
            ledState = !ledState;
            led.Write(ledState);
        }
    }
}
```

Obr. 10. Hlavní část programu pro FEZ Cerbuino Net GPIO

3.3 FEZ Cerbuino Net – Ethernet

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

Otevřít Ethernet komunikaci pro desku FEZ Cerbuino Net
Využít Ethernet komunikaci při vývoji aplikací

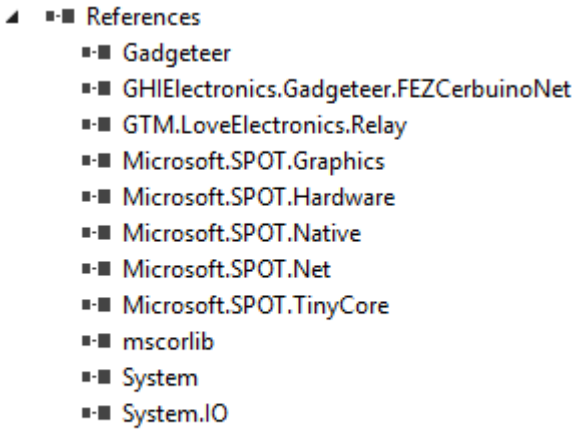
Výklad

Třetím programem je nastavení komunikace Ethernet, tento program má za cíl nejen aktivaci samotného Ethernet portu na desce, ale zároveň i jeho využití.

Program

Za pomocí kódu je na desce umožněna komunikace prostřednictvím LAN kabelu a poté je na desce vytvořen server pro komunikaci s počítačem, program obsahuje jeho inicializaci, proces a ukončení, zároveň obsahuje komunikační kód. Rovněž je k dispozici možnost vytvořit aplikaci umožňující počítači komunikovat s takto vytvořeným serverem. Důraz je kladen na umožnění samotné komunikace Ethernetem, kterou tato poskytnutá deska podporuje. Vzhledem k tomu, že je zvolena Gadgeteer aplikace z důvodu, že je díky strukturování zvolenému Visual Studiem snazší, je kód částečně odlišný od jiných dostupných materiálů. Na obrázku níže je zobrazena část kódu, která je zodpovídá za aktivaci Ethernetu a umožnění komunikace s počítačem, jako i reference a úvodní část programu. Další část kódu je značně rozsáhlá a obsahuje několik částí plnící funkce umožňující komunikaci s aplikací na straně počítače (jedná se o inicializace serverů, generování dat a různá ošetření).

```
using System;
using System.Collections;
using System.Threading;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Microsoft.SPOT.Presentation;
using Microsoft.SPOT.Presentation.Controls;
using Microsoft.SPOT.Presentation.Media;
using Microsoft.SPOT.Presentation.Shapes;
using Microsoft.SPOT.Touch;
using Microsoft.SPOT.Net;


using Gadgeteer.Networking;
using GT = Gadgeteer;
using GTM = Gadgeteer.Modules;
```



Obr. 11. Přehled využitých referencí a úvodní část programu

```
static NetworkInterface iface;
const Int32 Port = 2112;
IPEndPoint BindingEndPoint;
Socket Server = null;
Socket Client = null;
Thread ServerProcess = null;
Thread ClientCommunicationProcess = null;
GT.Timer timerServerStarted;        //Indikátor funkčního založení serveru
GT.Timer timerClientConnected;    //Indikátor navázání spojení, připojení klienta
int len;
bool LedOnOff = true;
// This method is run when the mainboard is powered up or reset.
void ProgramStarted()
{
    NetworkChange.NetworkAvailabilityChanged += new NetworkAvailabilityChangedEventHandler(NetworkChange_NetworkAvailabilityChanged);
    iface = NetworkInterface.GetAllNetworkInterfaces()[0] ;
    //iface.EnableStaticIP("192.168.0.101", "255.255.255.0", "");
    iface.EnableStaticIP("192.168.5.200", "255.255.255.0", "");

    timerServerStarted = new GT.Timer(1000); // every second (1000ms)
    timerServerStarted.Tick += new GT.Timer.TickEventHandler(timer_Tick);
    timerClientConnected = new GT.Timer(250); // every half second (500ms)
    timerClientConnected.Tick += new GT.Timer.TickEventHandler(timer_Tick);

    ServerStart(); //Pokus o založení serveru

    relay.SetRelay(Relay.AvailableRelays.Relay4, true);
    relay.SetRelay(Relay.AvailableRelays.Relay3, true);
    relay.SetRelay(Relay.AvailableRelays.Relay2, true);
    relay.SetRelay(Relay.AvailableRelays.Relay1, false);

    Mainboard.SetDebugLED(true); //Proces končí rozsvícením diody (je-li úspěšně server úspěšně založen, bude dioda blikat)
}

//Událost blikání diody
void timer_Tick(GT.Timer timer)
{
    Mainboard.SetDebugLED(LedOnOff = !LedOnOff);
}

//Událost fyzické změny stavu Sítě (Fyzická vrstva je/není k dispozici)
void NetworkChange_NetworkAvailabilityChanged(object sender, NetworkAvailabilityEventArgs e)
{
    if (e.IsAvailable)
    {
        Debug.Print("!! Network Available...");
        ServerStart();
    }
    else
    {
        Debug.Print("!! Network Unavailable...");
        ServerStop();
    }
}
```

Obr. 12. Hlavní část programu pro FEZ Cerbuino Net Ethernet

Shrnutí kapitoly

Kapitola poskytuje kódy k vytvoření základních programů pro desku FEZ Cerbuino Net, vysvětluje jejich funkci a poskytuje rozšiřující informace k těmto kódům.

Kontrolní otázka

1. Jak přidat reference.
2. Jaká z alternativ pro založení projektu je využívána.

Úkol k řešení

1. Vytvořit základní aplikaci pro desku
2. Vytvořit pokročilejší aplikaci s využitím referencí

4 ÚLOHY PRO STM32 F429 DISCOVERY

Po úspěšném a aktivním absolvování této KAPITOLY

Budete umět:	
Vytvořit několik aplikací pro STM32 F429 DISCOVERY Nahrát vytvořené aplikace na desku	Budete umět

Budete schopni:	
Vyzkoušet si spuštění aplikací na desce	Budete schopni

Kapitola popisuje dvě úlohy pro STM32 F429 DISCOVERY včetně kódů a znázornění nastavení pro nahrání programů pro desku. Je nezbytné následování předchozích částí návodů, jinak nebude Visual Studio schopno komunikovat s deskou.

4.1 STM32 F429 DISCOVERY – GPIO

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

Vytvořit jednoduchý program pro STM32 F429 DISCOVERY
Nahrát program na desku

Výklad

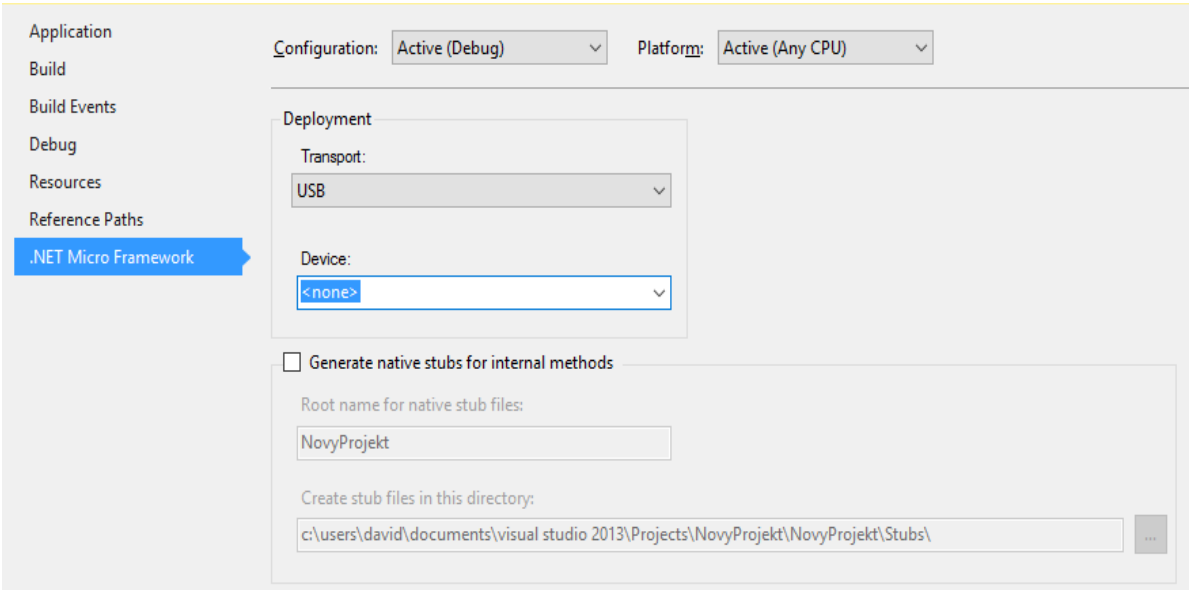
Jako první úloha pro desku STM32 F429 DISCOVERY je úloha s blikající LED, cíl představuje zejména seznámení se s funkcí desky, programovacím prostředím a rovněž jakým stylem jsou programy psány pro tuto desku obzvláště v této situaci, kdy.NET Micro Framework je nahrán ručně a není na desce od začátku.

Program

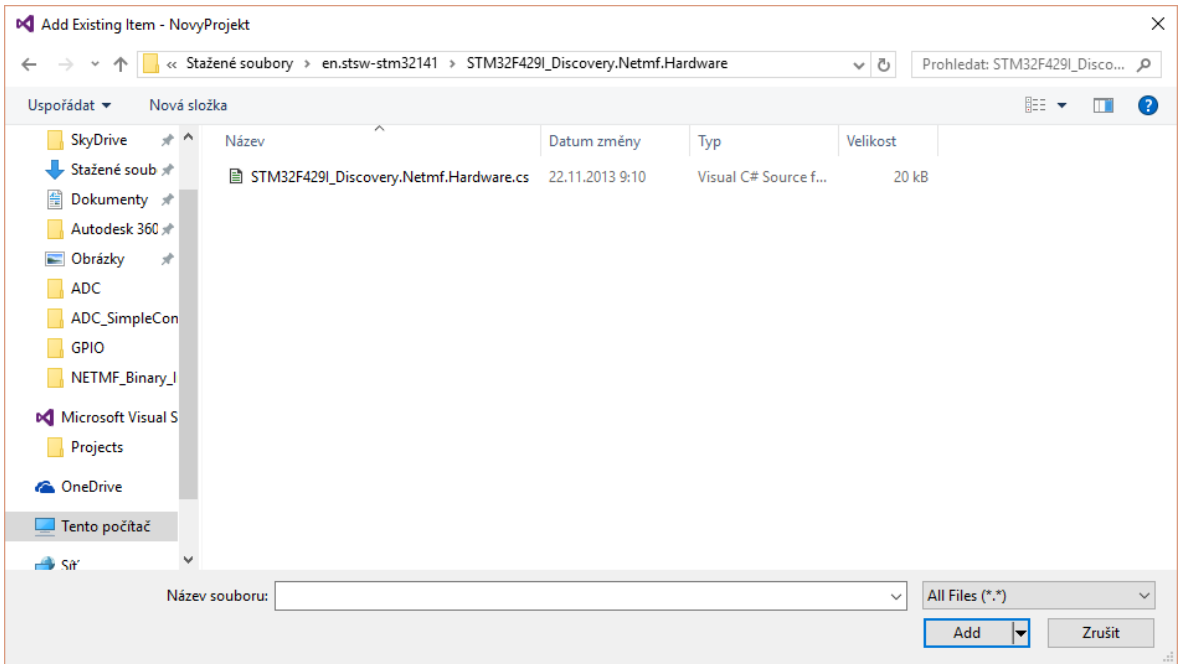
Programovaní je opět prováděno ve Visual Studiu a programovací jazyk je C#. Programy jsou tentokrát vytvořeny jako konzolové aplikace. Samotný program je jednoduchý a v případě následování výše zmíněných instrukcí by mělo sestavení programů být snadné. Program nesplňuje jiný účel než spuštění diody. Pro nahrání programu na desku je ještě nezbytné upravit nastavení ve Visual Studiu. Je třeba zvolit záložku Project, v bočním menu zvolit .NET Micro Framework a pod sekcí Deployment zvolit pro Transport možnost USB. Pokud jsme při konfiguraci USB komunikace pro desku postupovali správně, mělo by se pro Device automaticky zobrazit připojené zařízení. Pro správnou adresaci je rovněž nutné přidat STM32F429I_Discovery.Netmf.Hardware. Přidání se provede pravým kliknutím na název projektu v Solution Explorer, zvolit Add a možnost Existing Item a zvolit složku s takto pojmenovaným souborem.

```
using System.Threading;
using Microsoft.SPOT.Hardware;
using STM32F429I_Discovery.Netmf.Hardware;


public class BlinkingLed
{
    public static void Main()
    {
        LED.LEDInit(); // Inicializace LED GPIO

        while (true)
        {

            LED.RedLedToggle();
            Thread.Sleep(750);
            LED.GreenLedToggle();
            Thread.Sleep(750);

        }
    }
}
```

Obr. 13 Hlavní část programu pro STM32 F429 DISCOVERY GPIO



Obr. 14. Nastavení nahrání projektu na desku



Obr. 15. Lokalizace STM32F429I_Discovery.Netmf.Hardware

4.2 STM32 F429 DISCOVERY – ADC

Čas ke studiu: 1 hodina

Cíl Po prostudování tohoto odstavce budete umět

Vytvořit komplexnější program pro STM32 F429 DISCOVERY

Výklad

Dalším programem je A/D převodník, který je rozsahově jednoznačně větším projektem a má za cíl vyzkoušet další z možností desky.

Zapojení desky

Program nejprve načte hodnoty na analogových vstupech, podle desky můžeme naprogramovat několik vstupů, jež budou čteny programem, následně program vypíše hodnoty přečtené na vstupech do okna Output v programu Visual Studio, zároveň je opět inicializována LED, jež se zapne během cyklu, po jeho proběhnutí je zapnuto vyčkávání na 2 sekundy. Pro jeho nahrání na desku je opět třeba následovat stejného postupu jako u minulého projektu s nastavením komunikace přes USB ve vlastnostech projektu a opět nahrát STM32F429I_Discovery.Netmf.Hardware.

```
using System;
using Microsoft.SPOT;
using System.Threading;
using Microsoft.SPOT.Hardware;
using STM32F429I_Discovery.Netmf.Hardware;

namespace ADKonv
{
    public class Program
    {
        public static void Main()
        {
            AnalogInput ADC0 = new AnalogInput(ADC.Channel0_PA6);

            AnalogInput ADC1 = new AnalogInput(ADC.Channel1_PA7);

            AnalogInput ADC2 = new AnalogInput(ADC.Channel2_PC1);

            AnalogInput ADC3 = new AnalogInput(ADC.Channel3_PC3);

            /* Inicializace LED */
            LED.LEDInit();

            while (true)
            {
                /* Zobrazení ADC převedené hodnoty */
                Debug.Print("Channel0 (pin " + ADC0.Pin + ") = " + (ADC0.Read() * 3.3).ToString("f2") + "V");
                Debug.Print("Channel1 (pin " + ADC1.Pin + ") = " + (ADC1.Read() * 3.3).ToString("f2") + "V");
                Debug.Print("Channel2 (pin " + ADC2.Pin + ") = " + (ADC2.Read() * 3.3).ToString("f2") + "V");
                Debug.Print("Channel3 (pin " + ADC3.Pin + ") = " + (ADC3.Read() * 3.3).ToString("f2") + "V");


                Thread.Sleep(2000);

                LED.GreenLedToggle();
            }
        }
    }
}
```

Obr. 16. Hlavní část programu pro STM32 F429 DISCOVERY ADC

Shrnutí kapitoly

Kapitola se zabývá vytvářením programů pro STM32 F429 DISCOVERY a jejich nahráním na desku aby bylo možno sledovat jejich funkci.

Kontrolní otázka

1. Jak nastavit komunikaci Visual Studia s deskou.
2. Jak přidat STM32F429I_Discovery.Netmf.Hardware.

Úkol k řešení

1. Vytvořit základní aplikaci pro STM32 F429 DISCOVERY.
2. Vytvořit komplexnější program pro STM32 F429 DISCOVERY.